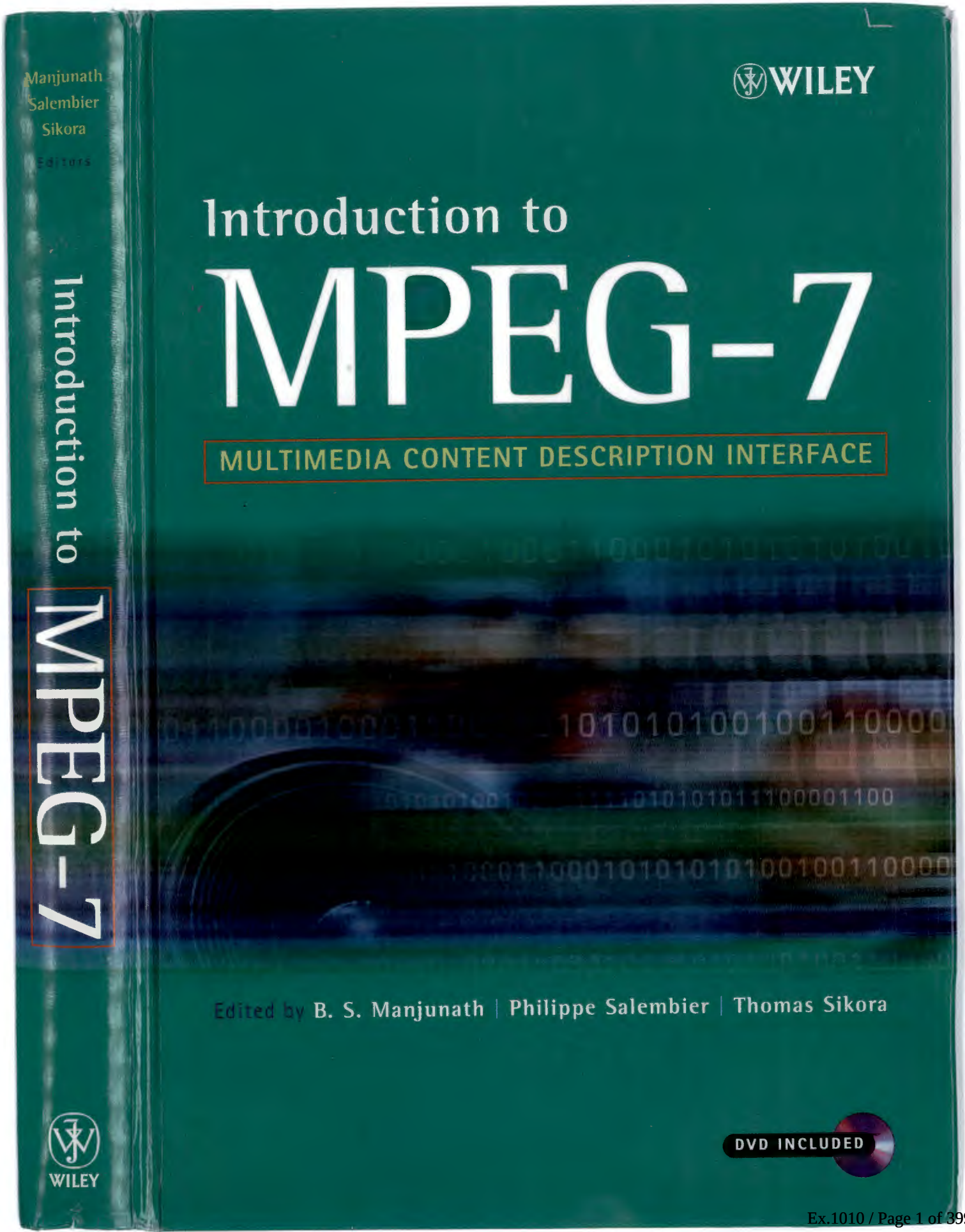

WILEY
Introduction to
MPEG-7
MULTIMEDIA CONTENT DESCRIPTION INTERFACE
Edited by B. S. Manjunath | Philippe Salembier | Thomas Sikora
DVD INCLUDED
Manjunath
Salembier
Sikora
Editors
Introduction to MPEG-7
WILEY

Introduction to MPEG-7

Multimedia Content Description Interface

Introduction to MPEG-7

Multimedia Content Description Interface

Edited by

B. S. Manjunath
University of California, Santa Barbara, USA

Philippe Salembier
Universitat Politècnica de Catalunya, Barcelona, Spain

Thomas Sikora
Heinrich-Hertz-Institute (HHI), Berlin, Germany



JOHN WILEY & SONS, LTD

Copyright © 2002 by John Wiley & Sons Ltd.,
Baffins Lane, Chichester,
West Sussex PO19 1UD, England

National 01243 779777
International (+44) 1243 779777
e-mail (for orders and customer service enquiries): cs-books@wiley.co.uk

Visit our Home Page on http://www.wileyeurope.com or http://www.wiley.com

All Rights Reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except under the terms of the Copyright, Designs and Patents Act 1988 or under the terms of a licence issued by the Copyright Licensing Agency, 90 Tottenham Court Road, London, UK W1P 9HE, UK, without the permission in writing of the Publisher, with the exception of any material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the publication.

The DVD included with this publication should not be sold or re-sold separately. In purchasing the Work, the purchaser acquires a non-exclusive licence for personal and non-commercial use of the DVD. The licence does not permit the reproduction or transmission of the contents of the DVD in any form or by any means without the written permission of the copyright holder. Requests for permission to reproduce material for any purpose, or to distribute it on a network, or to exploit it for any purpose other than personal, non-commercial use, should be addressed to the Permissions Department, John Wiley & Sons, Ltd, Baffins Lane, Chichester, West Sussex, PO19 1UD, UK or e-mailed to permreq@wiley.co.uk or faxed to +44 (0) 1243 770571.

The DVD may include utilities and programs in which the copyright is held by third parties. Wherever possible, the copyright holders are identified in the text files accompanying the utility or program. Requests for permission to exploit any software utilities or programs should be addressed to the appropriate copyright holder.

Other Wiley Editorial Offices

John Wiley & Sons, Inc., 605 Third Avenue,
New York, NY 10158-0012, USA

Wiley-VCH Verlag GmbH
Pappelallee 3, D-69469 Weinheim, Germany

John Wiley, Australia, Ltd, 33 Park Road, Milton,
Queensland 4064, Australia

John Wiley & Sons (Canada) Ltd, 22 Worcester Road
Rexdale, Ontario M9W 1L1, Canada

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01,
Jin Xing Distripark, Singapore 129809

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

ISBN 0 471 48678 7

Typeset in 10/12pt Times Roman by Laserwords Private Limited, Chennai, India.
Printed and bound in Great Britain by Antony Rowe Ltd., Chippenham, Wiltshire.
This book is printed on acid-free paper responsibly manufactured from sustainable forestry,
in which at least two trees are planted for each one used for paper production.

Contents

Contents of the DVD xv

Contributors xvii

Preface xxi

SECTION I INTRODUCTION 1

1 Introduction to MPEG-7: Multimedia Content Description Interface 3
Leonardo Chiariglione

2 Context, Goals and Procedures 7
Fernando Pereira and Rob Koenen
2.1 Motivation and Objectives 7
2.2 Driving Principles 9
2.3 What is Standardized? 10
2.4 MPEG Standards Development Process 11
2.5 MPEG-7 Types of Tools 13
2.6 Standard Organization and Workplan 15
2.7 Applications 17
2.8 Requirements 18
2.8.1 Requirements on Descriptors 19
2.8.2 Requirements on description schemes 19
2.8.3 Requirements on the DDL 19
2.8.4 Requirements on Descriptions 21
2.8.5 Requirements on Systems Tools 21
2.9 Conclusion 27
References 28

SECTION II SYSTEMS 31

3 Systems Architecture 33
Olivier Avaro and Philippe Salembier
3.1 Introduction 33
3.2 Objectives 34
3.2.1 Traditional MPEG Systems Requirements 34
3.2.2 MPEG-7 Specific Systems Requirements 35

3.3 MPEG-7 Terminal Architecture 35
3.3.1 Decoder Initialization 38
3.3.2 Processing of AUs 38
3.4 Action Units 39
3.5 Delivery of MPEG-7 Descriptions 40
3.6 Conclusion 40
Acknowledgment 40
References 41

4 Description Definition Language 43
Jane Hunter and Claude Seyrat
4.1 Introduction 43
4.2 Historical Background 44
4.3 XML Schema Structural Components 45
4.3.1 Namespaces and the Schema Wrapper 45
4.3.2 Element Declarations 46
4.3.3 Attribute Declarations 47
4.3.4 Type Definitions 47
4.3.5 Group Definitions 53
4.4 XML Schema Data Types 53
4.4.1 Built-in Primitive Data Types 54
4.4.2 Built-in Derived Data Types 55
4.4.3 Facets 56
4.4.4 The List Data Type 57
4.4.5 The Union Data Type 57
4.5 MPEG-7-Specific Extensions 58
4.5.1 Array and Matrix Data Types 58
4.5.2 Built-in Derived Data Types 59
4.6 Conclusion 59
References 60

5 Binary Format 61
Jörg Heuer, Cedric Thienot and Michael Wollborn
5.1 Overview 61
5.2 Fragment Update Command and Context 63
5.2.1 Fragment Update Command 64
5.2.2 Fragment Update Context 64
5.3 Binary Payload Representation 72
5.3.1 General Overview 73
5.3.2 Complex-Type Coding 74
5.3.3 Simple-Type Coding 77
5.3.4 Extensions and Forward/Backward Compatibility 77
5.4 Conclusion 79
Acknowledgment 80
References 80

SECTION III DESCRIPTION SCHEMES 81

6 Overview of Multimedia Description Schemes and Schema Tools 83
Philippe Salembier and John R. Smith
6.1 Introduction 83
6.1.1 Descriptors 83
6.1.2 Description Schemes 84
6.1.3 DDL 84
6.2 Organization of MDS Tools 84
6.2.1 Basic Elements 84
6.2.2 Content Management 86
6.2.3 Content Description 87
6.2.4 Navigation and Access 87
6.2.5 Content Organization 88
6.2.6 User Interaction 88
6.3 Schema Tools: *MPEG7* Root and Top-Level Elements 88
6.4 Conclusion 92
Acknowledgment 93
References 93

7 Basic Elements 95
Toby Walker, Jörg Heuer and José M. Martínez
7.1 Introduction 95
7.2 Basic Data Types 96
7.3 Linking and Localization Tools 97
7.4 Basic Tools 100
7.4.1 Relations and Graphs 101
7.4.2 Text Annotation 102
References 109

8 Description of a Single Multimedia Document 111
Ana B. Benitez, José M. Martinez, Hawley Rising and Philippe Salembier
8.1 Introduction 111
8.2 Content Management 112
8.2.1 Media Information 114
8.2.2 Content Creation 115
8.2.3 Content Usage 116
8.3 Content Structure 116
8.3.1 Segment Entities 117
8.3.2 Segment Attributes 120
8.3.3 Segment Decompositions 121
8.3.4 Structural Relations 124
8.4 Content Semantics 126
8.4.1 Abstraction Model 127

8.4.2 Semantic Entities 129
8.4.3 Semantic Attributes 132
8.4.4 Semantic Relations 134
8.4.5 Implicit and Explicit Semantic Description 136
8.5 Conclusion 138
References 138

9 Navigation and Summarization 139
Peter van Beek and John R. Smith
9.1 Introduction 139
9.2 Summarization 140
9.3 Views and View Decompositions 144
9.4 Variations 146
9.4.1 Automatic Extraction and Examples of Usage 148
9.4.2 Summarization 148
9.4.3 View Decompositions 149
9.4.4 Variations 149
9.5 Conclusion 150
References 150

10 Content Organization 153
John R. Smith and Ana B. Benitez
10.1 Introduction 153
10.2 Collections 155
10.3 Models 156
10.3.1 Probability Models 157
10.3.2 Analytic Models and Cluster Models 158
10.3.3 Classification Models 159
10.4 Examples of Usage 159
References 161

11 User Interaction 163
Peter van Beek, Kyoungro Yoon and A. Mufit Ferman
11.1 Introduction 163
11.2 Usage History 165
11.3 User Preferences 168
11.4 Examples of Usage 171
11.4.1 Mapping Usage History to User Preferences 171
11.4.2 Filtering Content Using User Preferences 173
11.5 Conclusion 174
References 174

SECTION IV VISUAL DESCRIPTORS 177

12 Overview of Visual Descriptors 179
B. S. Manjunath and Thomas Sikora
12.1 Introduction 179

12.2 Face Descriptor 181
12.2.1 Basis Vectors for the Face Images 181
12.2.2 Face Feature Extraction 182
12.3 A Quantitative Evaluation of Visual Descriptors 183
Acknowledgments 185
References 185

13 Color Descriptors **187**
Jens-Rainer Ohm, Leszek Cieplinski, Heon J. Kim, Santhana Krishnamachari, B. S. Manjunath, Dean S. Messing and Akio Yamada
13.1 Introduction 187
13.2 Color Spaces 189
13.2.1 HSV Color Space 189
13.2.2 HMMD Color Space 191
13.3 Dominant Color Descriptor 194
13.3.1 Extraction 195
13.3.2 Similarity Matching 196
13.3.3 Experimental Results 197
13.4 Scalable Color Descriptor 198
13.4.1 Extraction and Matching 199
13.4.2 Representation 200
13.4.3 Experimental Results 201
13.5 Group-of-Frame or Group-of-Picture Descriptor 201
13.5.1 Extraction and Matching 202
13.5.2 Descriptor Representation 202
13.5.3 Experimental Results 203
13.6 Color Structure Descriptor 204
13.6.1 CSD Interoperability 204
13.6.2 Extraction 205
13.6.3 CSD Resizing 208
13.6.4 Retrieval Results 208
13.7 Color Layout Descriptor 208
13.7.1 Extraction 209
13.7.2 Matching 210
13.7.3 Experimental Results 211
13.8 Summary 211
References 212

14 Texture Descriptors **213**
Yanglim Choi, Chee Sun Won, Yong Man Ro and B. S. Manjunath
14.1 Introduction 213
14.2 Homogeneous Texture Descriptor 214
14.2.1 Extraction 215
14.2.2 Experiments and Applications 217
14.3 Texture Browsing Descriptor 219
14.3.1 Definition and Semantics 219

14.3.2 Extraction 219
14.3.3 Regularity and Coarseness Estimation 220
14.3.4 Experiments and Applications 221
14.4 Edge Histogram Descriptor 223
14.4.1 Definition and Semantics 223
14.4.2 Extraction 223
14.4.3 Quantization 224
14.4.4 Matching 224
14.4.5 Experiments and Applications of the EHD 225
14.5 Summary 227
Acknowledgments 228
References 228

15 Shape Descriptors **231**
Miroslaw Bober, F. Preteux and Whoi-Yul Yura Kim
15.1 Introduction 231
15.2 Overview of Shape Descriptors 232
15.2.1 Region-Based Shape Descriptor 233
15.2.2 Contour-Based Shape Descriptor 233
15.2.3 3-D Shape Descriptor 234
15.2.4 Multiple-View Descriptor for Shape 234
15.3 Design and Evaluation of the Shape Descriptors 234
15.3.1 Evaluating Contour-Based Shape Descriptors 234
15.3.2 Testing Region-Based Shape Descriptors 237
15.3.3 Testing 3-D Shape Descriptors 237
15.4 Region-Based Shape Descriptor 238
15.4.1 ART Transform 238
15.4.2 Descriptor Representation 238
15.4.3 Similarity Measure 240
15.4.4 Experimental Results 240
15.5 Contour-Shape Descriptor 241
15.5.1 The CSS Representation 242
15.5.2 Descriptor Representation and Extraction 243
15.5.3 Properties of the Contour-Shape Descriptor 245
15.5.4 Experimental Results 245
15.5.5 Region-based vs Contour-based Shape Descriptors 246
15.5.6 Combining a Multiple-View and 2-D Shape Descriptors 247
15.6 3-D Shape Descriptor 247
15.6.1 The 3-D Shape Spectrum Descriptor 248
15.6.2 Syntax and Semantics of the 3-D SSD 251
15.6.3 Computation of the 3-D Spectrum Shape Descriptor 253
15.6.4 Example Similarity Measure 256
15.6.5 Experimental Results 256
15.7 Example applications of the MPEG-7 Shape Descriptor 256
15.7.1 Cartoon Search Engine 256
15.7.2 An Application of Region-Based Shape Descriptor for Retrieving Logos 258

15.8 Conclusion 258
Acknowledgments 259
References 260

16 Motion Descriptors **261**
Sylvie Jeannin, Ajay Divakaran and Benoit Mory
16.1 Introduction 261
16.2 Motion Basics 262
16.2.1 Motion Analysis 262
16.2.2 Motion Representation 264
16.3 Overall Organization of Motion Descriptions 265
16.3.1 Motion Characterization for Video Segments 266
16.3.2 Motion Characterization for Moving Regions 266
16.4 Motion Activity 267
16.4.1 Description 267
16.4.2 Extraction of Intensity of Motion Activity 268
16.4.3 Typical Usage 269
16.4.4 Further Possibilities and Discussion 269
16.5 Camera Motion 270
16.5.1 Description 270
16.5.2 Matching 272
16.6 Motion Trajectory 273
16.6.1 Description 273
16.6.2 Extraction 274
16.6.3 Usage 275
16.7 Parametric Motion 276
16.7.1 Description 276
16.7.2 Extraction 276
16.7.3 Usage 278
16.8 Conclusion 278
Acknowledgments 279
References 279

SECTION V AUDIO **281**

17 Fundamentals of Audio Descriptions **283**
Adam T. Lindsay, Ian Burnett, Schuyler Quackenbush and Melanie Jackson
17.1 The Structure of the Standard 283
17.2 Applications 283
17.2.1 Query by Humming 283
17.2.2 Query for Spoken Content 284
17.2.3 Extraction and Query Paradigm 284
17.2.4 Assisted Consumer-Level Audio Editing 285
17.3 Overview of Audio Descriptors 286
17.3.1 MPEG-7 Audio Description Framework 286
17.3.2 The LLD Interface 286

17.3.3 The Low-Level Audio Descriptors 289
17.3.4 High-Level Description Tools 295
17.3.5 Other Parts of the Standard 297
17.4 Summary 297
Acknowledgments 297
References 298

18 Spoken Content 299
J. P. A. Charlesworth and Philip N. Garner
18.1 Spoken Content in The Context of MPEG-7 299
18.1.1 Positioning Spoken Content Within MPEG-7 299
18.1.2 Importance of Spoken Content 300
18.2 ASR Technology Today 301
18.2.1 Drawbacks of ASR 301
18.2.2 Applications 301
18.3 Metadata Errors in Extraction – A Physical Argument 301
18.3.1 Quality of Audio 302
18.3.2 Quality of Speech 302
18.3.3 Quantity of Training Data 303
18.4 Interoperability Across Extraction Tools – A Technological Argument 303
18.5 Interoperability Across Databases 305
18.6 Structural Overview of the SpokencontentDescriptionScheme 306
18.6.1 SpokenContent Header 306
18.6.2 Word Lexicon 307
18.6.3 Phone Lexicon 308
18.6.4 Phone Confusion Statistics 309
18.6.5 Speaker Information 309
18.6.6 SpokenContent Lattice 310
18.7 Usage of the SpokenContentDescriptionScheme in an MPEG-7 Description 312
18.7.1 Header/Body Structure 312
18.7.2 Segmentation 312
18.7.3 Referencing 313
18.7.4 Binarization 313
18.8 Spoken Document Retrieval (SDR) 314
18.8.1 Literature 314
18.8.2 Example Evaluation 314
18.9 Summary 315
References 315

19 Sound Classification and Similarity 317
Michael A. Casey
19.1 Spectral Basis Functions 317
19.2 Sound Classification Models 320
19.3 Sound Probability Models 322

19.4 Training a Hidden Markov Model (HMM) 324
19.5 Indexing and Similarity Using Model States 325
19.6 Sound Model Applications 327
19.6.1 Automatic Audio Classification 327
19.6.2 Audio QBE 328
19.7 Summary 330
References 331

SECTION VI APPLICATIONS 333

20 Search and Browsing 335
Neil Day
20.1 Introduction 335
20.2 Growth of Rich Digital Content 336
20.3 MPEG-7 to the Rescue 336
20.3.1 Some MPEG-7 Application Scenarios 336
20.4 Real-Time Video Identification 338
20.4.1 Using MPEG-7 Tools 338
20.5 Query-By-Humming Application 339
20.5.1 Using MPEG-7 Tools 340
20.5.2 Observations: Variations in Implementation 341
20.6 Cuidado 342
20.6.1 Music Browser 342
20.6.2 Sound Palette 343
20.6.3 Use of MPEG-7 Description Tools 343
20.7 Television News Program Applications 344
20.7.1 News Document Model Based on MPEG-7 344
20.7.2 Filtering News Articles Using MPEG-7 345
20.8 Movie Tool 347
20.8.1 MPEG-7 Authoring Tool 347
20.9 TV-Anytime and MPEG-7 348
20.9.1 MPEG-7 Tools Used in TV-Anytime 349
20.10 Conclusion 350
Acknowledgment 350
References 351

21 Mobile Applications 353
Neil Day, Shun-ichi Sekiguchi and Mikio Sasaki
21.1 What you want, where you want, when you want. $(wyw)^3$ 353
21.2 Customized Multimedia Content Delivery for Mobile Users 354
21.2.1 An Application Scenario 354
21.2.2 Usage of MPEG-7 Tools 355
21.2.3 What Still Needs to be Done? 356
21.3 Real-time Information Retrieval for Mobile Users 357
21.3.1 Content Aggregation and Navigation 357
21.3.2 Utilization of Dependency 357

21.3.3 Application Scenario 358
21.3.4 What Still Needs to be Done? 359
21.4 Conclusion 359
Acknowledgments 360
References 360

Index **363**

Contents of the DVD*

Documents

- Additional reference documents are available on the DVD. These are marked with .

Demonstrations

- Columbia's Video Segment Retrieval and Browsing
- UCSB Browsing Aerial Images Using Texture
- ICU Homogeneous Texture
- Philips Video Shot Retrieval
- Mitsubishi Video Browser

Software

- MPEG-7 Reference Software
- BiM Software
- Schemas
- Schema Documentation (generated by Expway)
- Schema Documentation (generated by XMLSpy)

* The documents on the DVD are provided for informative purposes only. In many cases, the documents represent the views and conclusions of the authors and not of the MPEG. For the current status of the standard and consensus within the MPEG, the reader should refer to the published documents by the MPEG (see *www.cselt.it/mpeg*) and the ISO.

Contributors

Adam Lindsay
Computing Department Lancaster University
Lancaster
LA1 4YR
UK

Ajay Divakaran, Ph.D.
Mitsubishi Electric Research Laboratories
Murray Hill Laboratory
571 Central Avenue
Suite 115
Murray Hill, NJ 07974
ajayd@merl.com

Akio Yamada
Computer & Communication Research, NEC Corp.
Miyazaki 4-1-1, Miyamae,
Kawasaki 216-85555, Japan
a-yamada@da.jp.nec.com

Ana Belen Benitez
Electrical Engineering Department
Columbia University
1312 Mudd, #F6, 500 W. 120th St, MC 4712
New York, NY 10027
USA
Tel: +1 212 854-7473
Fax: +1 212 932-9421
ana@ee.columbia.edu

Benoit Mory
Laboratoires d'Electronique Philips
22 avenue Descartes, BP 15
94453 Limeil-Brevannes Cedex
France
benoit.mory@philips.com

Chi Sun Won
Department of Electrical Engineering
Dong Guk University
26 Beon-Ji, 3 Ga, Pil-Dong,
Joong-Gu, Seoul
South Korea
cswon@dgu.ac.kr

Claude Seyrat
Expway c/o Acland
18 avenue Georges V
75008 Paris
France
cseyrat@acland.fr

Cédric Thiénot
Expway c/o Acland
18 avenue Georges V
75005 Paris
France
cthienot@acland.fr

Dean S. Messing
Information Systems Technologies Dept.

Sharp Laboratories of America
5750 N.W. Pacific Rim Blvd.
Camas, WA, U.S.A. 98607
deanm@sharplabs.com

Fernando Manuel Bernardo Pereira,
Professor
Instituto Superior
Técnico – Instituto
de Telecomunicações
Av. Rovisco Pais, 1049-001
Lisboa, Portugal
Fernando.Pereira@lx.it.pt

Francoise Preteux
Institut National des
Télécommunications
Unité de Projets ARTEMIS
9, Rue Charles Fourier
91011 Evry Cedex – France
Francoise.Preteux@int-evry.fr

Hawley K. Rising III
Sony MediaSoft Lab, USRL
MD# SJ2C4
3300 Zanker Road
San Jose, CA 95134-1901
hawley.rising@am.sony.com

Heon Jun Kim
MI Group, Information
Technology Lab.
LG Electronics Institute of
Technology
16 Woomyeon-Dong, Seocho-Gu
Seoul, Korea 137-724
hjk@lge.co.kr

J. P. A. Charlesworth,
Reech Capital PLC,
1 Undershaft,
London EC3P 3DQ, England
jason_charlesworth@reech.com

Jane Hunter
DSTC Pty Ltd
Distributed Systems Technology
CRC
Level 7, General Purpose Sout
The University of Queensland
Queensland 4072
Australia
jane@dstc.edu.au

Jens-Rainer Ohm
Institute of Communication
Engineering
Aachen University of Technology
Melatener Str. 23, D-52072
Aachen, Germany
ohm@ient.rwth-aachen.de

John Smith
IBM T. J. Watson Research Center
30 Saw Mill River Road
Hawthorne, NY 10532
USA
jrsmith@watson.ibm.com

Jose M. Martinez
Grupo de Tratamiento de
Imagenes
Dpto. Señales, Sistemas y
Radiocomunicaciones
E.T.S.Ing. Telecomunicacion
(C-306)
Universidad Politècnica de Madrid
Ciudad Universitaria s/n
E-28040 Madrid
Spain
jms@gti.ssr.upm.es

Jörg Hueuer
Siemens AG
CT IC 2
Jörg Heuer
81730 München
Germany
Tel: +49 89 636 52957
Fax: +49 89 636 52393
Joerg.Heuer@mchp.siemens.de

Kyoungro Yoon
LG Electronics Institute of Technology
16 Woomyeon-dong, Seocho-gu,
Seoul 137-724
Korea
Tel: +82-2-526-4133
Fax: +82-2-526-4852
yoonk@lg-elite.com

Leonardo Chiariglione
Telecom Italia Lab
Via G. Reiss Romoli, 274
I-10148 Torino
Italy
leonardo.chiariglione@tilab.com

Leszek Cieplinski
Mitsubishi Electric ITE-VIL
20 Frederick Sanger Road
Guildford
Surrey
GU2 7YD
United Kingdom
Leszek.Cieplinski@vil.ite.mee.com

Michael Casey
M E R L
201 Broadway, 8th Floor
Cambridge MA 02139
mkc@merl.com

Michael Wollborn
Robert Bosch GmbH
FV/SLM
PO Box 777777
D-31132 Hildesheim
Germany
Michael.Wollborn@de.bosch.com

Mikio Sasaki
Research Laboratories, DENSO CORPORATION
500-1 Minamiyama,
Komenoki-cho, Nisshin-shi,
Aichi-ken,
470-0111 Japan
msasaki@rlab.denso.co.jp

Miroslaw Bober
Visual Information Laboratory
Mitsubishi Electric Information Technology Center Europe
20 Frederick Sanger Road
Guildford Surrey GU2 7YD, UK
miroslaw.bober@vil.ite.mee.com

Mufit Ferman
Sharp Laboratories of America
5750 N.W. Pacific Rim Blvd.
Camas, WA 98607
USA
mferman@sharplabs.com

Neil Day
MPEG-7 Alliance
Dublin, Ireland.
dneil@bluemetrix.com
Shun-ichi Sekiguchi
Multimedia Signal Processing Lab,
Multimedia Labs, NTT DoCoMo Inc.

Olivier Avaro
France Télécom R&D
38/40 rue Général Leclerc
92794 ISSY MOULINEAUX
Cedex 9
France
olivier.avaro@francetelecom.com

Peter van Beek
Sharp Labs of America,
5750 N.W. Pacific Rim Blvd.,
Camas, WA 98607
USA
Tel: +1 360-817-7622,
Fax: +1 360-817-8436,
pvanbeek@sharplabs.com

Philip N. Garner
Canon Research Centre
Europe Ltd,
1 Occam Court,
Occam Road,
Surrey Research Park,
Guildford,
Surrey GU2 7YJ.
United Kingdom
philg@cre.canon.co.uk

Philippe Salembier
Universitat Politècnica de
Catalunya
Campus Nord, Modulo D5
Jordi Girona, 1-3
08034 Barcelona,
Spain
Tel: +34 9 3401 7404
Fax: +34 9 3401 6447
philippe@gps.tsc.upc.es

Rob Koenen
InterTrust Technologies
Corporation
4750 Patrick Henry Drive
Santa Clara, CA 95054
USA
rkoenen@intertrust.com

Santhana Krishnamachari
Philips Research
345 Briarcliff Manor
New York 10510, USA
Santhana.krishnamachari@philips.com

Schuyler Quackenbush,
AT&T Labs, Rm E133 180 Park
Avenue, Bldg. 103 Florham Park,
NJ 07932, USA

Sylvie JEANNIN
Philips Research USA
345 Scarborough Road, Briarcliff
Manor NY 10510, USA

Thomas Sikora
Heinrich-Hertz-Institute for
Communication Technology
Einsteinufer 37 D-10587 Berlin
Germany
sikora@hhi.de

Toby Walker
Media Processing Division
Network and Software Technology
Center of America
3300 Zanker Road, MD #SJ2C4
San Jose, California 95134
tobyw@usrl.sony.com

Whoi-Yul Yura Kim
School of Electrical and Computer
Engineering
Hanyang University
Korea
wykim@email.hanyang.ac.kr

Yanglim Choi
Digital Media R&D Center,
Samsung Electronics Co., Ltd.
416, Maetan 3-Dong, Paldal-Gu,
Suwon, Kyungki-Do, S. Korea.
yanglimc@samsung.com

YongMan Ro
School of Engineering
Information and Communication
University
Yusong-Gu PO Box 77
Taejon, South Korea
yro@icu.ac.kr

Preface

This book provides a comprehensive introduction to the new ISO/MPEG-7 standard. The individual chapters are written by experts who have actively participated and contributed to the development of the standard. The chapters are organized in an intuitive way, with clear explanations of the underlying tools and technologies contributing to the standard. A large number of illustrations and working demonstrations should make this book a valuable resource for a wide spectrum of readers – from graduate students and researchers interested in the state-of-the-art media analysis technology to practicing engineers interested in implementing the standard.

SEARCH AND RETRIEVAL OF MULTIMEDIA DATA

Multimedia search and retrieval has become a very active research field because of the increasing amount of audiovisual (AV) data that is becoming available, and the growing difficulty to search, filter or manage such data. Furthermore, many new practical applications such as large-scale multimedia search engines on the Web, media asset management systems in corporations, AV broadcast servers, and personal media servers for consumers are about to be widely available. This context has led to the development of efficient processing tools that are able to create the description of AV material or to support the identification or retrieval of AV documents. Besides the research activity on processing tools, the need for interoperability between devices has also been recognized and several standardization activities have been launched. MPEG-7, also called "Multimedia Content Description Interface", standardizes the description of multimedia content supporting a wide range of applications. Standardization activities do not focus so much on processing tools but concentrate on the selection of features that have to be described, and on the way to structure and instantiate them with a common language.

As an emerging research area of wide interest, multimedia content description has a large audience. There are many workshops and conferences related to this topic every year, and their number is growing. The MPEG-7 technology covers the most recent developments in multimedia search and retrieval.

This book presents a comprehensive overview of the principles and concepts involved in a complete chain of AV material indexing, metadata description (based on the MPEG-7 standard), information retrieval and browsing. The book offers a practical step-by-step walk through of the components, from systems to schemas to audio–visual

descriptors. It addresses the selection of the multimedia features to be described, the organization and structuring of the description, the language to instantiate the description, as well as the major processing tools used for indexing and retrieval of images and video sequences. The accompanying electronic documentation will include numerous examples and working demonstrations of many of these components.

Researchers and students interested in multimedia database technology will find this book a valuable resource covering a broad overview of the current state of the art in search and retrieval. Practicing engineers in industry will find this book useful in building MPEG-7 compliant systems, as the only resource outside of the MPEG community available to the public at the time of publication.

ORGANIZATION

The book is organized into six sections: Introduction, Systems, Multimedia Description Schemes, Visual descriptors, Audio Descriptors and Applications.

Section I: Introduction

This section introduces the MPEG-7 standardization activity and the history behind this new standard. In Chapter 1, Leonardo Chiariglione, the convenor of MPEG, provides the motivation for the new standard. Chapter 2, by Pereira and Koenen, outlines the various activities within MPEG-7 that gained momentum towards the end of 1998, culminating in the final standard in 2002.

Section II: Systems

The systems section covers three major areas: Systems Architecture, Description Definition Language and the Binary Format for MPEG-7. The chapter on Systems Architecture discusses the design principles behind MPEG-7 Systems and highlights the most important processing steps for transport and consumption of MPEG-7 descriptions. The second chapter focuses on the language used to define the various description elements called Descriptors or Description Schemes that are presented in Sections III, IV and V. Finally, the Binary Format for MPEG-7 is described in Chapter 5. This format has been designed so as to efficiently compress and transport MPEG-7 descriptions.

Section III: Multimedia Description Schemes

Section III describes the organization of features that can be described with MPEG-7. The organization of this section is based on the functionality provided by the various Description Schemes. Chapter 6 provides an overview of the entire section. Chapter 7 discusses elementary Descriptions Schemes or Descriptors that are used as building blocks for more complex Descriptions Schemes. The tools available for description of a single multimedia document are reviewed in chapter 8. The most important features related to content management and description, including low-level as well as high-level features, are analyzed.

Purely audio or visual features are very briefly mentioned in this chapter. A detailed presentation of the corresponding set of tools is given in Sections IV (visual features) and Section V (audio features). The main functionalities supported by the tools of Chapter 8 include search, retrieval and filtering. Navigation and browsing are supported by a specific set of tools described in Chapter 9. Furthermore, the description of collections of documents or of descriptions is presented in Chapter 10. Finally, for some applications, it has been recognized that it is necessary to define in a normative way the user preferences and the usage history pertaining to the consumption of the multimedia material. This allows, for example, matching between user preferences and MPEG-7 content descriptions in order to facilitate personalization of the processing. These tools are described in Chapter 11.

Section IV: Visual Descriptors

This section begins with an overview in Chapters 12 and 13 describes color descriptors that represent different aspects of color distribution in images and video. These include descriptors for a color histogram of a single image as well as a collection of images, color structure, dominant color, and color layout. Chapter 14 presents three texture descriptors: a homogeneous texture descriptor, a coarse level browsing descriptor and an edge histogram descriptor. Chapter 15 presents descriptors that represent contour shape, region shape and 3-D shapes. The section concludes with motion descriptors in Chapter 16.

Section V: Audio Descriptors

An overview of the audio descriptors is provided in Chapter 17. Chapter 18 describes the spoken content technology in more detail. Sound recognition and sound similarity tools are outlined in Chapter 19.

Section VI: Applications

Finally, we conclude with a section on the potential applications of MPEG-7. The applications are broadly classified into search and browsing related, and mobile applications. Chapter 20 covers some interesting search and browsing applications that include real time video retrieval, browsing of TV news broadcast using MPEG-7 tools, and audio and music retrieval. Chapter 21 discusses two interesting mobile applications.

DVD

The accompanying DVD contains additional material, including technical reports, some working demonstrations and the official MPEG-7 reference software. The demonstrations on the DVD include video browsing and shot retrieval, and search and browsing of images using texture. We hope that researchers and graduate students will find this useful in their work.

ACKNOWLEDGMENTS

We would like to express our gratitude and sincere thanks to all the contributors without whose dedication and timely contributions this work would not have been possible. Our special thanks to Leonardo Chiariglione, the convenor of MPEG, for his encouragement and support throughout the course of this project. We would like to thank the International Organization for Standardization (ISO) and in particular Jacques-Olivier Chabot and Keith Brannon for allowing us to publish the MPEG-7 Reference software on the accompanying DVD.

We would also like to thank Dr. Lutz Ihlenburg of Heinrich-Hertz-Institut, Germany, for assisting on editorial issues and for providing many valuable comments and suggestions. Our thanks to Shawn Newsam and Lei Wang for organizing the material for the DVD. We extend our thanks to the many reviewers who helped editing individual chapters.

BSM would also like to thank Samsung Electronics for its support in facilitating the participation in the MPEG-7 activities. Special thanks to Dr. Hyundoo Shin and Dr. Yanglim Choi for their support during the past three years. Thanks to Shawn Newsam, Xinding Sun, Gomathi Sankar, Ying Li, Ashish Agarwal, and Lei Wang for reviewing some of the chapters. He would like to thank all the members of the vision research laboratory at UCSB for their help in putting together this manuscript.

B. S. Manjunath
Philippe Salembier
Thomas Sikora

Section I
INTRODUCTION

1

Introduction to MPEG-7: Multimedia Content Description Interface

Leonardo Chiariglione

Telecom Italia Lab, Italy

Established in 1988, the Moving Picture Experts Group (MPEG) has developed digital audiovisual compression standards that have changed the way audiovisual content is produced by manifold industries, delivered through all sorts of distribution channels and consumed by a variety of devices.

MPEG is intimately connected to digital audio and video. However, when MPEG made its first breath, bits were already abundant, but they were kind of 'heavy' bits. They were part of Pulse Code Modulation (PCM) samples of music stored on compact discs. No one thought of storing or moving a song around in digital form when this meant to store or move 50 MB, unless this was done in a special environment like a studio. The only known way of moving audio and video was in the shape of analog waveforms.

MPEG-1 and MPEG-2 changed this environment radically. Audio files became manageable, the more so if the user was willing to get the music with some artifacts in exchange of a reduced file size or reduced transmission time. The number of television programs started multiplying by orders of magnitude. First, because more television programs in digital form could be packed in the bandwidth that used to carry one television program and second, because of the ability to make new offerings, thanks to the new economies of scale made possible by audio and video in digital form. Compact discs could be used to store movies, and new types of compact discs were even invented to store movies in new forms. Then MPEG-4 extended the possibility to deliver audio and video to new environments such as the Internet and mobile channels.

With the full range of bit-rates covered by the three standards, it could have been expected that MPEG follow the advice of an interested party: to take a sabbatical or, better, to simply disband. The latter option leaves the field open to anybody claiming: 'My algorithm performs 1.5 times better than MPEG' (typical claim of a conference paper) or 'My algorithm performs 3 times better than MPEG' (typical claim of a naive salesman) or 'My algorithm performs 10 times better than MPEG' (typical claim of a

confessed liar). As a corollary, the MPEG Convenor could have followed another piece of advice from the same interested party and look after his vineyard.

Not so. Already in November 1993, when MPEG was still busy finalizing MPEG-2 and working to define the scope of MPEG-4, a proposal had been made to Subcommittee 29 of JTC 1 (the parent body of MPEG). This proposal requested that MPEG investigate the need to develop a standard that would allow users to identify the content that would be present in '500 channels' (The slogan of those times that represented the effect of digitization of television on the business of Community Antenna Television, CATV). That was the beginning of the idea, but MPEG-7 or 'Multimedia Content Description Interface' only started officially in 1997.

Having completed MPEG-2, at least the most crucial parts of its audio-video-systems parts of it, MPEG could mobilize its energies to plan its work beyond MPEG-4. That work turned out to be the ideal continuation of the work proposed in 1993, that is, the definition of an audiovisual information representation standard that would describe or express the semantic meaning of the information and therefore enable people to discover what is in a set of audiovisual objects without having to actually access the information itself.

New MPEG standards have always presented cultural challenges. The development of MPEG-1 provided the first opportunity for people from different industry backgrounds, mostly consumer electronics and telecommunications, to come together and develop a coding standard. Probably the most innovative concept, both from the technical and business viewpoint, was the clear separation of what would constitute the standard – the decoder – and what was left unspecified and open to competition – the encoder. MPEG-2 was the opportunity for a massive intake of people from the television world, both broadcasters and consumer electronics, to join together and make the resulting team work successfully. At the time of MPEG-4, more people from 'new' industry segments came to MPEG to develop the multimedia and mobile standard.

In the new 'MPEG-7 community' that gradually built up, it was not clear what the difference was between what was needed in an algorithm or an implementation and what was required in a standard. It was also not clear which were the interfaces that the standard made reference to. Even less clear was what characterized an 'MPEG-7 encoder' and which were the exact functions that were left for 'encoder optimization' and which were the subject of standardization because they made reference to the 'MPEG-7 decoder'. The role of the textual representation versus binary representation was also not clear because MPEG-7 applications were not just of the communication type in which binary encoding is paramount for practical applications but were also for local database applications for which binary coding is not necessarily a requirement. The demarcation between Audio or Visual Descriptors and Description Schemes seemed impossible to find.

The reason was that MPEG-7 is, conceptually, an audiovisual information representation standard, but the representation satisfies very specific requirements.

The difficulties highlighted notwithstanding, MPEG-7 has turned out to be a very solid and effective standard. The Audio and Video parts provide standardized 'audio only' and 'visual only' descriptors, the Multimedia Description Schemes (MDS) part provides standardized description schemes involving both audio and visual descriptors,

the Description Definition Language (DDL) provides a standardized language to express description schemes, and the Systems part provides the necessary glue that enables the use of the standard in practical environments. Lastly, the Reference Software contains the huge contributions made by the community to develop the MPEG-7 'Open Source' code.

MPEG-7 will fulfill a key function in the forthcoming evolutionary steps of multimedia. As much as MPEG-1, MPEG-2 and MPEG-4 provided the tools through which the current abundance of audiovisual content could happen, MPEG-7 will provide the means to navigate through this wealth of content. MPEG-7 is not the only shop in town in this area. However, the other 'metadata' initiatives have all been developed to serve the specific needs of one business environment. But this is the age in which content companies are by no means constrained by their own traditional delivery mechanisms. In the same way, content consumers are no longer tied to a single source of content. This is the real value of MPEG-7, a value that is in line with past MPEG standard of providing a generic solution that is technically agnostic of the environment.

This book has been written by some of the most authoritative individuals playing key roles in the development of the MPEG-7 standard. Even though this book is not a substitute to the standard itself, it provides a general overview, as well as insights, into the critical elements of the standard and information that is pertinent to understanding MPEG-7 and its practical use.

2

Context, Goals and Procedures

Fernando Pereira and Rob Koenen

Instituto Superior Técnico, Lisbon, Portugal, InterTrust Technologies Corp., CA, USA

2.1 MOTIVATION AND OBJECTIVES

Producing multimedia content today is easier than ever before. Using digital cameras, personal computers and the Internet, virtually every individual in the world is a potential content producer, capable of creating content that can be easily distributed and published. The same technologies allow content, which would in the past remain inaccessible, to be made available on-line.

However, what would seem like a dream can easily turn into an ugly nightmare if no means are available to manage the explosion in available content. Content, analogue and digital alike, has value only if it can be discovered and used. Content that cannot be easily found is like content that does not exist, and potential revenues are directly dependent on users finding the content. The easier it becomes to produce content, the faster the amount of content grows and the more complex the problem of managing content gets. The same digital technology that lowers the thresholds for producing and publishing content can also help in analyzing and classifying it, in extracting and manipulating features for specific applications and in searching and discovering content. Be it with or without automated support, information about content is a prerequisite for being able to find and manage it.

To date, people looking for content have used text-based browsers with very moderate retrieval performance; typically, these search engines yield much noise around the hits. The fact that they are in widespread use nonetheless indicates that a need exists. These text-based engines rely on human operators to manually describe the multimedia content with keywords and free annotations. For two reasons this is increasingly unacceptable. First, it is a costly process, and the cost increases with the growing amount of content. Second, these descriptions are inherently subjective and their usage is often confined to the application domain that the descriptions were created for. Hence, it is necessary to automatically and objectively describe, index and annotate multimedia information, notably audiovisual data, using tools that automatically extract (possibly complex) audiovisual features from the content to substitute or complement manual, text-based

descriptions. These automatically extracted audiovisual features will have three advantages over human annotations: (1) they will be automatically generated, (2) they can be more objective and domain-independent and (3) they can be native to the audiovisual content. Native descriptions would use nontextual data to describe content, using features such as color, shape, texture, melody and sound envelopes, in a way that allows the user to search by comparing descriptions. Even though automatically extracted descriptions will be very useful, it is evident that descriptions, the 'bits about the bits', will always include textual components. There are many features that can only be expressed through text, for example, authors and titles.

The situation depicted above has been recognized for a number of years now, and a lot of work has been invested in the last years in researching relevant technologies. Several products addressing this problem have already emerged in the market, such as Virage's Videologger [1]. These products, as well as the large number of papers in journals, conferences and workshops, were an indication that the time was ripe to address the multimedia content description problem at a much larger scale.

The aforementioned problem and the technological situation were recognized by MPEG (the Moving Picture Experts Group) [2] in July 1996, when it decided, at the Tampere MPEG meeting, to start a standardization project, generally known as MPEG-7, and formally called *Multimedia Content Description Interface* (ISO/IEC 15938) [3]. The MPEG-7 project has the objective to specify a standard way of describing various types of multimedia information: elementary pieces, complete works and repositories, irrespective of their representation format and storage medium. The objective is to facilitate the quick and efficient identification of interesting and relevant information and the efficient management of that information [4]. These descriptions are both textual (annotations, names of actors etc.) and nontextual (statistical features, camera parameters etc.). Like the other members of the MPEG family, MPEG-7 defines a standard representation of multimedia information satisfying a set of well-defined requirements. But MPEG-7 is quite a different standard than its predecessors. MPEG-1, MPEG-2 and MPEG-4 all represent the content itself – 'the bits' – while MPEG-7 represents information about the content – 'the bits about the bits'. While the first reproduce the content, the latter describes the content. The requirements for these two purposes are very different [5], although there is also some interesting overlap in technologies, and sometimes the frontiers are not that sharp.

Even without MPEG-7, there are many ways to describe multimedia content in use today in various digital asset management systems. Such systems, however, generally do not allow a search across different repositories and do not facilitate content exchange between different databases using different description systems. These are interoperability issues, and creating a standard is an appropriate way to address them. A standard way to do multimedia content description allows content and its descriptions to be exchanged across different systems. Also, it sets an environment in which tools from different providers can work together, creating an infrastructure for transparent management of multimedia content. The main results of the MPEG-7 standard are this increased interoperability, the prospect to offer lower-cost products through the creation of a sizable market with new, standard-based services and a rapidly growing user base [6]. This agreement – a standard is no more and no less than an agreement between its users – will stimulate both content providers and users and simplify the entire content-identification process. Of course, the standard needs to be technically sound, since otherwise proprietary solutions will prevail,

which will hamper interoperability. The challenge in MPEG-7 was matching the needs with the available technologies, or, in other words, reconciling what is possible with what is useful.

Participants in the development of MPEG-7 represent broadcasters, equipment and software manufacturers, digital content creators and managers, telecommunication service providers, publishers and intellectual property rights managers, as well as university researchers.

2.2 DRIVING PRINCIPLES

The MPEG-7 standardization project was preceded by an exploration phase, in which some fundamental principles appeared to be generally shared among all participants [4]. These driving principles are more than just high-level requirements, as they express the vision behind MPEG-7. This vision has guided the requirements gathering process [5] and the subsequent tools development work [4].

The guiding principles, which set the foundations of the MPEG-7 standard, are as follows [4]:

- *Wide application base*: MPEG-7 shall be applicable to the content associated to any application domain, real-time-generated or not; MPEG-7 shall not be tuned to any specific type of application. Moreover, the content may be stored, and may be made available on-line, off-line or streamed.
- *Relation with content*: MPEG-7 shall allow the creation of descriptions to be used:
  - stand-alone, for example, just providing a summary of the content;
  - multiplexed with the content itself, for example, when broadcast together with the content;
  - linked to one or more versions of the content, for example, in Internet-based media.
- *Wide array of data types*: MPEG-7 shall consider a large variety of data types (or modalities) such as speech, audio, image, video, graphics, 3-D models, synthetic audio and so on. Since MPEG-7 emphasis is on audiovisual information, no new description tools should be developed for textual data. Rather, existing solutions shall be considered such as Standard Generalized Markup Language (SGML), Extensible Markup Language (XML) or Resource Description Framework (RDF) [5].
- *Media independence*: MPEG-7 shall be applicable independently of the medium that carries the content. Media can include paper, film, tape, CD, a hard disk, a digital broadcast, Internet streaming and so on.
- *Object-based*: MPEG-7 shall allow the object-based description of content. The content can be represented, in this case described, as a composition of multimedia objects and it shall be possible to independently access the descriptive data regarding specific objects in the content.
- *Format independence*: MPEG-7 shall be applicable independently of the content representation format, whether analogue or digital, compressed or uncompressed. Therefore, audiovisual content could be represented in Phase Alternate Line (PAL), National Television Standards Committee (NTSC), MPEG-1, MPEG-2 or MPEG-4 and so forth. There is, however, a special relation with MPEG-4 [7, 8] since both MPEG-7 and

MPEG-4 are multimedia representation standards that are built using an object-based data model. As such, they are both unique and they complement each other very well, allowing very powerful applications to be created.

- *Abstraction level*: MPEG-7 shall include description capabilities with different levels of abstraction, from low-level, often statistical features, to high-level features conveying semantic meaning. Often the low-level features can be extracted automatically, whereas the more semantically meaningful features need to be extracted manually or semiautomatically. Also, different levels of description granularity shall be possible within each abstraction level. Note that higher-level conclusions often find evidence in lower-level features.
- *Extensibility*: MPEG-7 shall allow the extension of the core set of description tools in a standard way. It is recognized that a standard such as MPEG-7 can never contain all the structures needed to address every single application domain, and thus it shall be possible to extend the standard in a way that guarantees as much interoperability as possible.

These principles not only characterize the MPEG-7 vision but they also indicate what sets MPEG-7 apart from other similar standardization efforts.

2.3 WHAT IS STANDARDIZED?

Technology and standards, as so many things in life, may get old and obsolete. In addition, since the less flexible and dynamic they are, the easier it is for them to become obsolete, it is essential that standards are as flexible and minimally constraining as possible, while still serving their fundamental objective – interoperability. To MPEG, this means that a standard must specify the minimum necessary, but not more than that. This approach allows industrial competition and further evolution of the technology in the so-called 'nonnormative' areas – the areas that the standard does not fix. To MPEG-7, this implies that only the description format – syntax and semantics – and its decoding will be standardized. Elements that are explicitly not specified are techniques for extraction and encoding, and the 'consumption' (description usage) phase. Although good analysis and retrieval tools will be as essential for a successful MPEG-7 application as motion estimation and rate control are for MPEG-1 and MPEG-2 applications, and video segmentation for some MPEG-4 applications, their standardization is not required for interoperability; in fact, the descriptions' consumer does not care that much about the way the descriptions are created, provided that they can be understood and used. The specification of content analysis tools – automatic or semiautomatic – is out of the scope of the standard, and so are the programs and machines that 'consume' MPEG-7 descriptions. Developing these tools will be a task for the industries that build and sell MPEG-7-enabled products. This approach ensures that good use can be made of the continuous improvements in the relevant technical areas. New technological developments can be leveraged to build improved automatic analysis tools, matching engines and so on, and the descriptions they produce or consume will remain compliant with the standard. Therefore, progress need not stop at the moment the standard is frozen. It is possible to rely on technical competition for obtaining ever better results. This is happening for MPEG-2, where improvements in encoding

techniques have slashed bit-rates for digital television almost to half over the past four years.

The first edition of the MPEG-7 standard is commonly designated as Version 1. The standard will be extended in the future with additional tools to address more requirements and provide more functionality. This will happen in the form of amendments to the standard. It is common to designate as Version N of a part of the standard, the set of tools in Version 1 extended with the tools specified in Amendment N-1 for that part of the standard; for example, Amendment 1 of a part of the MPEG-4 standard is commonly known as the Version 2 of that part of the standard.

2.4 MPEG STANDARDS DEVELOPMENT PROCESS

When content representation changed from analogue to digital, the technology development process also changed, if only in terms of speed of developments and the fact that it was no longer sufficient to simply designate vertical columns of technology for well-defined applications. Thus, it is essential for standardization bodies such as MPEG to take this environment into account in the standards it creates and the way it sets those standards. For a decade now, it has no longer been possible to employ the 'system-driven approach' in which the value of a standard is limited to a specific, vertically integrated system. Modern standards shall ensure interoperability across countries, services and applications [9].

Hence, MPEG takes a horizontal approach to standardization rather than a vertical one. It uses a toolbox philosophy, in which an MPEG standard shall provide a minimum set of relevant tools, which, after being assembled according to industry needs, provide the maximum interoperability at a minimum complexity and cost.

The success of MPEG standards is mainly based on this toolbox approach, combined with the 'one functionality, one tool' principle [9]. This principle means that MPEG standards shall not include dual tools for achieving the same functionality. In summary, MPEG wants to offer its users interoperability and flexibility, at the lowest complexity and cost.

To ensure a timely and technically sound development of standards, the development process is organized as follows:

Requirements phase

1. Applications: Identify relevant applications using input from the MPEG members; inform potential new participants about the new *upcoming* standard (note that new participants are always welcome in MPEG; in practice, many new people join at the onset of a new standard).

2. Functionalities: Identify the functionalities needed by the applications above.

3. Requirements: Describe the requirements following from the functionalities above in such a way that common requirements can be identified for different applications; identify which requirements are common across the areas of interest and which are not common but still relevant.

Development phase

1. Call for proposals: A public call for proposals is issued, asking all interested parties to submit technology that could fulfill the identified requirements.

2. Evaluation of proposals: The proposals are evaluated in a well-defined, adequate and fair evaluation process, which is published with the call itself. The process may entail, for example, subjective testing, objective comparison or evaluation by experts.

3. Technology selection: As a result of the evaluation, the technologies best addressing the requirements are selected. MPEG does not usually choose one single proposal, but typically starts by assembling a framework that uses the best-ranking proposals, combining those. This is the start of a collaborative process to draft and improve the standard.

4. Collaborative development: The collaboration includes the definition and improvement of a 'Working Model', which embodies early versions of the standard and can include nonnormative parts. The Working Model evolves by having alternative tools challenge those already in the Working Model, by using the so-called 'Core Experiments (CEs)'. In MPEG-7, the Working Model is called *eXperimentation Model* (XM). Core Experiments are technical experiments according to predefined conditions carried out by multiple independent parties. Their results are the basis for technological choices.

5. Balloting: When a certain level of maturity has been achieved, national standardization bodies review the Draft Standard in a number of ballot rounds, voting to promote the standard and asking for changes.

Verification phase

1. Verification: Verify that the tools developed can be used to assemble the target systems and provide the desired functionalities. This is done by means of 'Verification Tests'. For MPEG-1 through 4, these tests were mostly subjective evaluations of the decoded quality. For MPEG-7, they have to assess efficiency in identifying the right content described using MPEG-7 tools.

This development process is not rigid; some steps may be repeated and iterations are sometimes needed. Requirements collection is typically an activity that extends throughout the development process, as new requirements become apparent over time, often as a result of building the standard. The time schedule is, however, always closely observed by MPEG. All decisions within the MPEG Working Group (WG) are taken by consensus; nevertheless, the process keeps a high pace, allowing MPEG to timely provide good technical solutions. Keeping the established deadlines is very important to allow the industry to plan products that build on the standard.

The period until the evaluation of the proposals is referred to as the 'competitive phase', while the period after the evaluation is the 'collaborative phase'. During the collaborative phase, all the MPEG members collectively improve and complete the most promising tools identified at the proposals' evaluation. The collaborative phase is the major strength of the MPEG process, as hundreds of the best experts in the world, from over a hundred companies and universities, work together toward a common goal. So it should not come as a surprise that this superteam traditionally achieves excellent technical results, which in turn justifies the effort of the participating companies to participate, or, if that is not feasible, at least follow the process.

Two working tools play a major role in the collaborative development phase that follows the initial competitive phase: the MPEG-7 XM and CE [10]. It is important to realize that the Core Experiments will not end up in the standard itself, as these are just working tools to ease the development phase.

1. XM: The MPEG-7 XM is a complete framework, such that an experiment performed by multiple independent parties will produce essentially identical results. The XM enables the checking of the relative performance of different tools, as well as improving the performance of selected tools.

The XM is built after screening the proposals answering the call for proposals. The XM is not the best proposal but a combination of the best tools, independent of the proposal they belong to. The XM includes components for evaluating and improving the various types of description tools. It typically contains normative and nonnormative tools to create the complete 'common framework', which allows performing adequate evaluation and comparison of tools targeting the continuous improvement of the technology included in the XM. After the first XM is established, that is, already in the collaborative phase, new tools can be brought to MPEG-7 after evaluation inside the XM following a core experiment procedure. The XM evolves through versions as core experiments verify the advantages of including new tools or prove that included tools should be substituted. At each XM version, only the best performing tools are part of the XM. If any part of a proposal is selected for inclusion in the XM, the proposer must provide the corresponding source code for integration into the XM software. The XM evolved into the Reference Software [11], which is a normative part of the MPEG-7 Standard.

2. Core experiments: The improvement of the XM started with a first set of core experiments, which were defined at the conclusion of the evaluation of proposals. Core experiments are multiple, independent, directly comparable experiments that determine whether or not a proposed tool has merit. Proposed tools may target the substitution of a tool in the XM or the direct inclusion in the XM to provide a new, relevant functionality not yet present. Improvements and additions to the XM are based on the results of core experiments.

A core experiment has to be completely and uniquely defined, so that the results are unambiguous. In addition to the specification of the tool to be evaluated, a core experiment also specifies the conditions to be used, again so that the results can be compared. One or more MPEG experts propose a core experiment, which has to be accepted by consensus, provided that two or more independent experts agree to perform the experiment. Proposers who find their tools accepted for inclusion in the XM must provide the corresponding source code for the XM software. So far, many tens of core experiments have been performed to select the tools adopted for the standard.

2.5 MPEG-7 TYPES OF TOOLS

Rather early in the development of MPEG-7, it was clear that a number of different tools were needed to achieve the standard's objectives. These tools are descriptors (the elements), description schemes (the structures), a Description Definition Language (DDL) (for extending the predefined set of tools) and a number of Systems tools.

More precise definitions of these tools, taken from the MPEG-7 Requirements document [5], are presented in the following:

- Descriptors (D)
  - A descriptor is a representation of a feature, where a feature is a distinctive characteristic of the data[1] that signifies something to somebody.
  - A descriptor defines the syntax and the semantics of the feature representation.
  - A descriptor allows an evaluation of the corresponding feature via the descriptor value[2]. It is possible to have several descriptors representing a single feature, for example, to address different relevant requirements/functionalities.
  - A descriptor corresponds to attributes at the level of the MPEG-7 Audiovisual Conceptual Model [5][3].
  - A descriptor does not participate in many-to-one relationships with other description elements.
  - *Examples*: a time code for representing duration, color moments and histograms for representing color and a character string for representing a title.
- Description Schemes
  - A description scheme specifies the structure and semantics of the relationships between its components, which may be both descriptors and description schemes.
  - A description scheme provides a solution to model and describe multimedia content in terms of structure and semantics.
  - A description scheme corresponds to an entity or relationship at the level of the MPEG-7 Audiovisual Conceptual Model [5].
  - A description scheme shall have descriptive information and may participate in many-to-one relationships with other description elements.
  - A simple example is a movie, temporally structured as scenes and shots, including some textual descriptors at the scene level and color, motion and audio amplitude descriptors at the shot level.

[1] Data is audiovisual information that will be described using MPEG-7, regardless of storage, coding, display, transmission, medium, or technology [5].

[2] A descriptor value is an instantiation of a descriptor for a given data set (or subset thereof) [5].

[3] The MPEG-7 Conceptual Model provides a model of the audiovisual domain at the conceptual level [5]. The Conceptual Model is independent of the design or implementation of the MPEG-7 description schemes and descriptors in the MPEG-7 Description Definition Language. The MPEG-7 Conceptual Model employs the following modeling constructs:

- Entity – A principal object in the audiovisual domain;
- Relationship – An association among one or more entities as follows: (1) Association – relates two or more entities which do not exhibit existence dependency; (2) Generalization – specifies an "is-A" relationship that partitions a class of entities into mutually exclusive subclasses, and (3) Aggregation – specifies a "has-A" assembly-component relationship of entities;
- Attribute – Descriptive information about an entity or relationship that can be used for identification or description.

The MPEG-7 Conceptual Model also defines each principal concept in words and assigns a domain and type to each concept. The domain information indicates whether the concept relates specifically to audio or visual content, or generically applies to the audiovisual domain.

- Description Definition Language (DDL)
  - The DDL is a language that allows the creation of new description schemes and, possibly, descriptors.
  - The DDL also allows the extension and modification of existing description schemes.
- Systems tools
  - Tools related to the binarization, synchronization, transport and storage of descriptions, as well as to the management and protection of intellectual property.

These are the 'normative tools' of the MPEG-7 standard. 'Normative' means here that if these tools are implemented, they must be implemented according to the standardized specification, as these tools are essential to guarantee interoperability.

With these tools, MPEG-7 description 'creators' can build MPEG-7 descriptions. A *description* consists of one or more instantiated description schemes, which together describe the data at hand. A description scheme in a *description* can be fully or partly instantiated, which means that values may be attributed to all or just some of the descriptors [5].

2.6 STANDARD ORGANIZATION AND WORKPLAN

The MPEG-7 standard is organized in several parts. This was done to allow the various clusters of technology to be used stand-alone, according to the MPEG's toolbox principles. They may even be used in conjunction with proprietary technologies. Another reason for having different parts is that the editing of the standard remains manageable.

Examples of stand-alone usage of parts are also found, for example, in MPEG-2 Video, which is used together with MPEG-2 Systems but not with MPEG-2 Audio in the US digital television system. While the various parts of MPEG-7 may be used independently or in combination with proprietary technologies, they were developed as a consistent and coherent whole, with the most powerful applications using them together.

The MPEG-7 standard is structured in eight parts as follows [4]:

- *Part 1 – Systems*: Specifies the tools that are needed to prepare MPEG-7 descriptions for efficient transport and storage (binarization), to allow synchronization between content and descriptions and the tools related to managing and protecting intellectual property [12].
- *Part 2 – Description Definition Language*: Specifies the language for defining new description schemes (and possibly also new descriptors[4]) [13].
- *Part 3 – Visual*: Specifies the descriptors and description schemes dealing exclusively with visual information [14].
- *Part 4 – Audio*: Specifies the descriptors and description schemes dealing exclusively with audio information [15].

[4] Defining new descriptors is not yet possible in MPEG-7 Version 1, available by the summer 2001.

- *Part 5 – Generic Entities and Multimedia Description Schemes (MDS)*: Specifies the descriptors and description schemes dealing with generic (nonaudio or video-specific) and multimedia features [16].
- *Part 6 – Reference Software*: Includes software corresponding to the tools included in the standard[5] [11].
- *Part 7 – Conformance Testing*: Defines guidelines and procedures for testing conformance of MPEG-7 descriptions and terminals [17].
- *Part 8 – Extraction and Use of MPEG-7 Descriptions*: Provides information on the extraction and use of some description tools, notably giving insight into the Reference Software [18]. This part is a Technical Report and not a Standard. It basically corresponds to the textual version of the Visual part of the XM, which describes all the normative and nonnormative visual tools implemented in the XM software.

Parts 1 to 5 specify the core MPEG-7 technology, while parts 6 to 8 are 'supporting parts'. The schedule adopted for MPEG-7 is presented in Table 2.1 – MPEG-7 Workplan. Addition of tools to the various parts of the MPEG-7 standard may be done through amendments. Amendments are also referred as new versions, where the first edition of the standard is called Version 1, for example, Amendment 1 to part 1 of the standard is Version 2 for that part.

After an initial period dedicated to the specification of objectives and the identification of applications [19] and requirements [5], MPEG issued, in October 1998, a MPEG-7 Call for Proposals [20]. By the December 1st, 1998 deadline, 665 proposal preregistrations were received [21]. Of these, 390 (59%) were actually submitted as proposals by

Table 2.1 MPEG-7 Workplan

October 98	Call for proposals Final version of the MPEG-7 Proposal Package Description (PPD)
December 98	Preregistration of proposals
February 1, 99	Proposals due
February 15–19, 99	Evaluation of proposals (ad hoc group meeting held in Lancaster, UK)
March 99	First version of the MPEG-7 XM
December 99	Working Draft stage (WD)
October 00	Committee Draft stage (CD)
March 01	Final Committee Draft stage (FCD) after ballot with comments
July 01	Final Draft International Standard stage (FDIS) after ballot with comments (after this step, the text of the standard is not subject to changes)
September 01	International Standard (IS) after yes/no ballot
May 2002[a]	Proposed Draft Amendment (PDAM) 1 to parts 1, 3, 4, 5 and 6
October 2002	Final Proposed Draft Amendment (FPDAM) 1 to parts 1, 3, 4, 5 and 6 after ballot with comments
March 2003	Final Draft Amendment (FDAM) 1 to parts 1, 3, 4, 5 and 6 after ballot with comments
May 2003	Amendment (AMD) 1 to parts 1, 3, 4, 5 and 6 after yes/no ballot

[a]The workplan for MPEG-7 Version 2 (Amendment 1) is tentative.

[5]ISO grants a free license to the copyright of the code to whoever wants to implement compliant products; this does not mean that product developers do not have to license the required patent licenses.

the February 1st, 1999 deadline. Out of these 390 proposals, there were 231 descriptors and 116 description schemes proposals. The proposals for normative elements were evaluated by MPEG experts, in February 1999, in Lancaster (UK), following the procedures defined in the MPEG-7 Evaluation document [22]. A special set of audiovisual content was provided to the proposers for usage in the evaluation process; this content has also been used in the collaborative phase. The content set consists of 32 compact discs with sound tracks, pictures and moving video [23], made available to MPEG under the licensing conditions defined in Reference 24. Broadly speaking, these licensing terms permit usage of the content exclusively for MPEG-7 standard development purposes.

While fairly straightforward methodologies were used for the evaluation of the audiovisual description tools in the MPEG-7 competitive phase, more powerful methodologies were developed during the collaborative phase in the context of the tens of core experiments performed. After the evaluation of the technology received, choices and recommendations were made and the collaborative phase started with the most promising tools [25]. In the course of developing the standard, additional calls may be issued when not sufficient technology is available within MPEG to meet the requirements, but there must be indications that the technology does indeed exist outside MPEG; otherwise, MPEG may decide to develop by itself the necessary tools following a collaborative approach.

Table 2.1 shows that the development of the first version of the MPEG-7 standard took about three years between issuing the first Call for Proposals and the publication of the International Standard. MPEG is generally known for its very challenging schedules, but creating a standard like this takes considerable time. This time has to be weighed against the probabilities of getting a default, proprietary standard, if 'official' standardization comes too late. This schedule also does not take into account the undefined additional time that the companies owning the essential patents necessary to implement the standard will take to set the licensing procedure in order so that the industry may start selling products based on that standard.

2.7 APPLICATIONS

As demonstrated above, MPEG-7's requirements are derived from the relevant applications, which were identified first. The applications relevant for MPEG-7 may be both new applications as well as existing ones, with no difference in priority in shaping the standard. Since the MPEG-7 standard needs to be generic, many different application domains that may benefit from the MPEG-7 standard were identified, which makes it difficult to come up with an exhaustive application list.

The MPEG-7 Applications document [19] lists the following application domains:

Education	Surveillance
Journalism	Biomedical applications
Tourist information	Shopping
Cultural services	Architecture, real estate, interior design
Entertainment	Social
Investigation services, forensics	Film, video and radio archives
Geographic information systems	Audiovisual content production
Remote sensing	

The MPEG-7 Applications document includes examples of both applications that can be improved by using the MPEG-7 standard as well as new ones that may be enabled by it. The document organizes the example applications into three sets [19]:

- *Pull applications*: Applications such as storage and retrieval in audiovisual databases, delivery of pictures and video for professional media production, commercial musical applications, sound effects libraries, historical speech database, movie scene retrieval by memorable auditory events and registration and retrieval of trademarks.
- *Push applications*: Applications such as user agent–driven media selection and filtering, personalized television services, intelligent multimedia presentations and information access facilities for people with special needs.
- *Specialized professional applications*: Applications that are particularly related to a specific professional environment, notably teleshopping, biomedical, remote sensing, educational and surveillance applications.

For each application, the MPEG-7 Applications document gives a description of the application, the corresponding requirements and a list of related relevant work and references. It is understood that the set of applications in the document is not complete; the intent is just to give the industry – clients of the MPEG work – some suggestions. It is expected that MPEG-7 will enable new, unforeseen applications. MPEG-7 is very likely the MPEG standard with the finest implementation granularity: it is possible to build an MPEG-7 application with very few MPEG-7 tools. Owing to this fact, and the nature of the standard in general, MPEG-7 applications will very probably hit the market soon after the standard is final – much closer to finalization than with the MPEG-4 standard.

2.8 REQUIREMENTS

The requirements gathering started early in the development process, and has continued ever since [5]. The requirements are the basis of the development process. Tools are developed to fulfill the identified requirements, and technology that fulfills no requirements is not considered.

Requirements have been extracted from the identified applications [19]. The MPEG-7 requirements [5] are divided into five categories: descriptors, description schemes, DDL, descriptions and Systems requirements. The collective set of MPEG-7 tools must satisfy all the requirements. Individual tools need not and cannot fulfill them all. It is the right combination of tools that allows building the algorithms that are able to address the specific needs of a certain class of applications. Whenever applicable, visual and audio requirements are considered separately. The requirements apply, in principle, to both real time and non–real time as well as to off-line and streaming applications.

The MPEG-7 Requirements document includes, for each requirement, a definition of the requirement as well as some notes and examples relevant to understand the requirement in question. The requirements are listed below, organized on concise tables (Tables 2.2 to 2.6). Expanding them further would go beyond the scope of this book[6]. To

[6] The list of requirements reflects the situation as of July 2001.

Table 2.2 Requirements on descriptors

Name	Definition
Cross-modality	MPEG-7 descriptors shall support audio, visual or other descriptors which allow queries based on visual descriptions to retrieve audio data and vice versa.
Direct data manipulation	MPEG-7 descriptors shall be able to act as handles referring directly to the data, to allow manipulation of the multimedia material.
Data adaptation	MPEG-7 descriptors shall allow the transcoding, translation, summarization and adaptation of multimedia material according to the capabilities of client devices, network resources, user and author preferences and user environments.
Language of text-based descriptions	MPEG-7 text descriptors shall specify the language used in the description. MPEG-7 text descriptors shall support all natural languages.
Linking	MPEG-7 descriptors shall support a mechanism allowing source data to be located in space and in time using the MPEG-7 data descriptors. MPEG-7 shall also support a mechanism to link to related information.
Prioritization of related information	MPEG-7 descriptors shall support a mechanism allowing the prioritization of related information, mentioned under Linking above.
Unique identification	MPEG-7 descriptors shall support a mechanism for uniquely identifying data, such that it provides an unambiguous method for associating descriptions with the described data.

get all the details about the MPEG-7 requirements, the reader should consult the MPEG-7 Requirements document [5]. It should be clear that not all the requirements have the same priority and not all have had an equal impact on the MPEG-7 work.

2.8.1 Requirements on Descriptors

This section includes the requirements on descriptors, see Table 2.2 [5].

2.8.2 Requirements on description schemes

This section includes the requirements on description schemes, see Table 2.3 [5].

2.8.3 Requirements on the DDL

This section includes the requirements for the DDL [5]. As it will be seen in the chapter dedicated to the DDL, the MPEG-7 DDL is based on W3C's XML Schema Language; some extensions to XML Schema had to be developed by MPEG in order to fulfill all the DDL requirements presented in Table 2.4 – requirements on the Description Definition Language.

Table 2.3 Requirements on Description Schemes

Name	Definition
Description scheme relationships	MPEG-7 description schemes shall express the relationships between descriptors to allow the use of the descriptors in more than one description scheme. The capability to encode equivalence relationships between descriptors in different description schemes shall also be supported.
Prioritization of descriptors	MPEG-7 description schemes shall support the prioritization of descriptors in order that queries may be processed more efficiently. The priorities may reflect levels of confidence or reliability.
Hierarchy of descriptors	MPEG-7 description schemes shall support the hierarchical representation of different descriptors in order that queries may be processed more efficiently in successive levels where N level descriptors complement (N – 1) level descriptors.
Scalability of descriptors	MPEG-7 description schemes shall support scalable descriptors in order that queries may be processed more efficiently in successive description layers. An N-layer description is an enhancement or refinement of an (N – 1) layer description.
Description of temporal range	MPEG-7 description schemes shall support the association of descriptors to different temporal ranges, both hierarchically (descriptors are associated to the whole data or a temporal subset of it) and sequentially (descriptors are successively associated to successive time periods).
Data adaptation	MPEG-7 description schemes shall allow the transcoding, translation, summarization and adaptation of multimedia material according to the capabilities of the client devices, network resources, user and author preferences and user environments.

Table 2.4 Requirements on the Description Definition Language

Name	Definition
Compositional capabilities	The DDL shall allow new description schemes and descriptors to be created and existing description schemes to be modified or extended.
Unique identification	The DDL shall allow a unique identification of descriptors and description schemes.
Primitive data types	The DDL shall provide a set of primitive data types, e.g., text, integer, real, date, time/time index, version etc.
Composite data types	The DDL shall be able to describe composite data types such as histograms, graphs, RGB values, enumerated types etc.
Multiple media types	The DDL shall provide a mechanism to relate descriptors to data of multiple media types of inherent structure, particularly audio, video, audiovisual presentations, the interface to textual description and any combinations of these.
Various types of description scheme instantiations	The DDL shall allow various types of description scheme instantiation: full, partial, full-mandatory, partial-mandatory.
Relationships within a description scheme and between description schemes	The DDL shall be able to express spatial, temporal, structural and conceptual relationships between the elements of a description scheme and between description schemes.

Table 2.4 (*continued*)

Name	Definition
Relationship between description and data	The DDL shall supply a rich model for links and references between one or more descriptions and the data described.
Link to ontologies	The DDL shall supply a linking mechanism between a description and several ontologies.
Platform independence	The DDL shall be platform- and application-independent.
Grammar	The DDL shall follow an unambiguous and easily parsed grammar.
Validation of constraints	A DDL parser shall be capable of validating: (1) values of properties, (2) structures, (3) related classes and (4) values of properties of related classes.
Human readability	The DDL shall allow descriptors and description schemes to be readable by humans.
Real-time support	The DDL shall support real-time applications.
Forward and backward compatibility	The DDL shall allow backward compatibility meaning that descriptions created in any MPEG-7 version shall always remain valid in later versions. The DDL should allow forward compatibility meaning that Version 1 parsers or decoders need to be able to understand the Version 1 part of descriptions created using MPEG-7 as extended in future versions.[7]

2.8.4 Requirements on Descriptions

This section includes the requirements on descriptions, see Table 2.5. These requirements have common aspects to the requirements for descriptors, description schemes and the DDL. Note that descriptions are obtained by instantiating one or more description schemes and are themselves not defined by the standard.

2.8.5 Requirements on Systems Tools

This section lists the MPEG-7 Systems requirements, see Table 2.6. MPEG-7 descriptions can be represented either in textual format – using the DDL, in binary format – using the BiM (Binary format for MPEG-7 data) or a mixture of the two formats, depending on the application [4]. MPEG-7 defines a unique bidirectional mapping between the binary and the textual formats; this mapping can be lossless either way.

Checking whether all the requirements have been met is an ongoing process, much like updating the requirements. Not all requirements will be met in Version 1; some of them are for Version 2 and others may be removed if no technical work is developed to address them.

[7] This requirement is stated as a "should" because its implications are not yet fully understood.

Table 2.5 Requirements on Descriptions

Name	Definition
General requirements	
Types of features	MPEG-7 shall support multimedia descriptions using various types of features.
Abstraction levels for multimedia material	MPEG-7 shall support hierarchical mechanisms to describe multimedia documents at different levels of abstraction. This supports users' needs for information at differing levels of abstraction.
Management of descriptions	MPEG-7 shall support the ability to manage multiple descriptions and partial descriptions of the same material at several stages of its production process, as well as descriptions that apply to multiple copies of the same material. It shall be possible to share the descriptions among several clients for simultaneous access and controlled modification.
Translations in text descriptions	MPEG-7 textual descriptions shall provide a way to contain translations into a number of different languages. It shall be possible to convey the relation between the descriptions in the different languages.
Associated information	MPEG-7 shall support other information associated with the data.
Referencing analogue data	MPEG-7 descriptions shall provide the ability to reference and describe multimedia documents in analogue format.
Associate relations	MPEG-7 shall support relations between components of a description.
Functional requirements:	
Retrieval effectiveness	MPEG-7 shall support the effective retrieval of multimedia material.[8]
Retrieval efficiency	MPEG-7 shall support efficient retrieval of multimedia material.[9]
Similarity-based retrieval	MPEG-7 shall support descriptions allowing rank-ordering content by the degree of similarity with the query.
Streamed and stored descriptions	MPEG-7 shall support both streamed (synchronized with content) and nonstreamed data descriptions.
Distributed multimedia databases	MPEG-7 shall support the simultaneous and transparent retrieval of multimedia data in distributed databases.
Interactive queries	MPEG-7 shall support mechanisms to allow interactive queries.
Browsing	MPEG-7 shall support descriptions allowing to preview information content in order to aid users to overcome their unfamiliarity with the structure and/or types of information or to clarify their undecided needs.
Interactivity support	MPEG-7 shall support the means allowing specifying the interactivity related to a description.
User preferences	MPEG-7 shall support the means to specify user's preferences in browsing, filtering and searching multimedia material, respecting the user's privacy.

[8] The user gets what he is looking for and not other material.
[9] The user gets what he is looking for, quickly.

Table 2.5 (*continued*)

Name	Definition
User usage history	MPEG-7 shall support the means to specify user's usage history in browsing, filtering and searching multimedia material, respecting the user's privacy.
Key items	MPEG-7 shall support mechanisms to indicate the most relevant parts of a description, which are called key items.
Ordering keys	MPEG-7 shall support mechanisms to indicate the most relevant descriptors for ordering subsets of a certain type of description information, such as key items.
Temporal validity	MPEG-7 shall support a mechanism to specify the temporal interval (start time and duration) of the validity or usefulness of descriptions.
Coding requirements:	
Description-efficient representation	MPEG-7 shall support the efficient representation of data descriptions.
Description extraction	MPEG-7 shall standardize descriptors and description schemes that are easily extractable from uncompressed and compressed data, according to several widely used formats.[10]
Robustness to information errors and loss	MPEG-7 shall provide mechanisms that guarantee graceful behavior of the MPEG-7 system in the case of transmission errors.
Visual-specific requirements:	
Types of features	MPEG-7 shall at least support visual descriptions allowing the following features (mainly related to the type of information used in the queries): color, visual objects, texture, sketch, shape, still and moving images, volume, spatial relations, motion, deformation, source of visual object and its characteristics, models, 'encoding decisions'.
Data visualization using the description	MPEG-7 shall support a range of multimedia data descriptions with increasing capabilities in terms of visualization. This means that MPEG-7 data descriptions shall allow a more or less sketchy visualization of the indexed data.
Visual data formats	MPEG-7 shall support the description of the following visual data formats: digital video and film, analogue video and film, still pictures in electronic, paper or other format, graphics, 3-D models, composition data associated to video and others to be defined.
Visual data classes	MPEG-7 shall support descriptions specifically applicable to the following classes of visual data: natural video, still pictures, graphics, animation (2-D), 3-D models, composition information.

(*continued overleaf*)

[10] The fact that there is a requirement on easy feature extraction does not imply that the extraction method is to be standardized but just that the method has to be simple.

Table 2.5 *(continued)*

Name	Definition
Audio-specific requirements:	
Types of features	MPEG-7 shall support audio descriptions allowing the following features (mainly related to the type of information used in the queries): frequency contour, audio objects, timbre, harmony, frequency profile, amplitude envelope, temporal structure (including rhythm), textual content, sonic approximations, prototypical sound, spatial structure, source of sound and its characteristics and models.
Data sonification using the description	MPEG-7 shall support a range of multimedia data descriptions with increasing capabilities in terms of sonification.
Auditory data formats	MPEG-7 shall support at least the description of the following types of auditory data: digital audio, analogue audio, Musical Instrument Digital Interface (MIDI) audio, model-based audio and production data.
Auditory data classes	MPEG-7 shall support descriptions specifically applicable to the following subclasses of auditory data: sound track (natural audio scene), music, atomic sound effects, speech, symbolic audio representations, mixing information.
Text-specific requirements:[11]	
Text retrieval	The adopted text description tools and their interfaces shall allow queries based on audiovisual descriptions to retrieve text data and vice versa.
Consistency of text description tools	MPEG-7 text description tools for text-only documents and composite documents containing text should have the same meaning that the descriptors and description schemes for text-only documents and for text in visual documents (e.g., subtitles) must have.

Table 2.6 Requirements on Systems Tools

Name	Definition
General requirements:	
Multiplexing of descriptions	MPEG-7 shall support: (1) embedding multiple MPEG-7 descriptions into a single data stream and (2) embedding MPEG-7 description(s) into a single data stream together with associated content.
Flexible access to partial descriptions at the systems level	MPEG-7 shall support the efficient selection of partial descriptions without the need of decoding the full description.
Temporal synchronization of content with descriptions	MPEG-7 shall support the temporal association of descriptions with content that can vary over time.

[11] Since MPEG-7's emphasis is on audiovisual content, providing novel solutions to describe text is not among the goals of MPEG-7. Although MPEG-7 will consider existing solutions developed by other standardization organizations to describe text documents, and support them as appropriate; these are the requirements on such solutions.

Table 2.6 *(continued)*

Name	Definition
Synchronization of multiple descriptions over different physical locations	If multiple descriptions of a content exist, MPEG-7 shall support mechanisms to keep them consistent.
Physical location of content with associated descriptions	MPEG-7 shall support the association of descriptions with content that can vary in physical location.
Transmission mechanisms for MPEG-7 streams	MPEG-7 shall support the transmission of MPEG-7 descriptions using a variety of transmission protocols.
MPEG-7 file format	The MPEG-7 file format shall support flexible user-defined access to parts of the description(s) that it contains. Linking mechanisms with other files or described content objects should be provided.
Robustness to information errors and loss	MPEG-7 shall support mechanisms that guarantee graceful behavior of the MPEG-7 system in the case of transmission errors.
Quality of Service (QoS)	MPEG-7 shall support mechanisms for defining the Quality of Service (QoS) for the transmission of descriptions.
Carried descriptions	MPEG-7 shall provide for the fragmentation and reassembling of descriptions in an unambiguous way, targeting a variety of transport means, including the carriage of descriptions over MPEG-2.
Partition of descriptions	MPEG-7 shall provide a partitioned representation for MPEG-7 descriptions, in both textual and compact binary format. The partitioned representation shall allow streaming, incremental transfer and storage of MPEG-7 descriptions and schema. The textual and corresponding binary partitioned representations shall be semantically equivalent.
Efficient parsing	MPEG-7 shall enable efficient parsing in the binary and textual domain. The partitioned representation shall be easily wrapped into various transport layers providing underlying services such as bitstream resynchronization, random access points and delineation of packets. The partitioned representation shall enable the checking of syntactical correctness of an MPEG-7 stream in the binary and textual domain.
Efficient updating of descriptions	MPEG-7 shall provide a fast and efficient way to update an MPEG-7 description, and in this context enable at least the following two minimal functionalities: add and delete a subtree in a description tree.
Timed updating	MPEG-7 shall enable timed updates of MPEG-7 descriptions.
Intellectual property management and protection requirements:	
No legal status of descriptions	MPEG-7 descriptions by nature have no legal bearing. MPEG-7 shall be designed in such a way that this is as clear as possible.
Describing content rights	MPEG-7 shall provide a mechanism for pointing to content rights (rights owners, contractual information). Descriptors and description schemes shall not directly describe content rights.

(continued overleaf)

Table 2.6 (*continued*)

Name	Definition
Relationship to content management and protection measures	MPEG-7 shall accommodate, and not by design interfere with, rights management information and technological protection measures used to manage and protect content.
Applications distinguishing between legitimate and illegitimate content	MPEG-7 shall support applications that distinguish between legitimate and illegitimate content. MPEG-7 shall be constructed so as to allow clear and unambiguous reference, in external specifications, agreements and in legislation, to the clauses in the MPEG-7 standard addressing the requirement on 'No legal status of descriptions' above.
Authentication of descriptions	MPEG-7 shall offer a mechanism to allow for the authentication of MPEG-7 descriptions.
Management and protection of descriptions	MPEG-7 shall support the management of intellectual property in descriptions and protection from unauthorized access, use and modification.
Management and protection of descriptors and description schemes	MPEG-7 shall support the management of intellectual property in descriptors and description schemes and protection from unauthorized access, use and modification.
Usage rules	MPEG-7 shall contain descriptors and description schemes that provide information on how content may be used.
Usage history	MPEG-7 shall contain descriptors and description schemes that provide information on how content has been used, in accordance with privacy rules.
Identification of content	MPEG-7 shall enable the identification of content by international identification conventions.
Identification of content in descriptions	Where the description contains content, MPEG-7 shall enable the identification of that content by international identification conventions.
Identification of descriptions	MPEG-7 may need to enable the unique identification of descriptions.[12]
Binary representation (BiM) requirements:	
Compactness	The BiM shall provide a compact binary and dynamic representation for MPEG-7 descriptions.
Streaming, transfer and storage	The BiM shall allow streaming, incremental transfer and storage of descriptions.
Parsing	The BiM shall allow fast parsing of descriptions.
Applicability to individual descriptors and DSs	The BiM shall be applicable to individual descriptors and description schemes and it shall allow the binary representation of dynamically defined description schemes, i.e. schemas and/or instances.
Mapping with the DDL	The BiM shall be equivalent to the textual description (DDL) and it shall provide a unidirectional and a bidirectional mapping between the BiM and the DDL. If different solutions for these two mappings provide different efficiency regarding storage or transmission bandwidth and parsing speed, the BiM shall be defined in two levels, each level providing maximum efficiency for one of the mappings.

[12] This requirement was under study in July 2001.

Table 2.6 *(continued)*

Name	Definition
Well-formedness and validation	The BiM shall be designed in a way that allows a BiM parser to check the syntactical correctness, i.e., well formedness, and the validity regarding the normative aspects of an MPEG-7 description.
Easy wrapping	The BiM shall be easily wrapped into various transport layers providing underlying services such as bitstream resynchronization, random access points and delineation of packets.
Textual representation[13] *requirements*:	
Description tree updating	The textual representation shall provide a fast and efficient way to update a description tree.
Adding and deletion	The textual representation shall allow at least the following two minimal functionalities: add and delete an element in a description tree.
Scheduling of update executions	The textual representation shall allow the scheduling of update executions.
Description tree coherency	The textual representation shall keep the coherency of the description tree.

As explained in describing the MPEG standards development process, the requirements' phase is followed by technical work: comparing, selecting and optimizing the tool set. The remaining chapters of this book will address those tools.

2.9 CONCLUSION

MPEG-1 and MPEG-2 have been successful standards that have enabled widely adopted commercial products, such as CD-interactive products, digital audio broadcasting and digital television. Still, these standards are limited to the linear functionalities inherent to their underlying data representation model. The MPEG-4 standard adopts an object-based representation approach, modeling a scene as a composition of objects, both natural and synthetic, with which the user may interact. This opened new frontiers to the way users will play with, create, reuse, access and consume multimedia content.

With the MPEG-7 standard, MPEG moves from the area of data, content or essence to the area of the descriptions or metadata. MPEG-7 tackles the issues involved in managing content, including searching, selecting and filtering it.

In comparison with other available or emerging solutions for multimedia description, MPEG-7 can be characterized by

- its genericity, related to its capability to consistently describe content from many application domains;

[13] These requirements may, in the future, be included as DDL requirements.

- the integration of low-level and high-level descriptors into a single architecture, allowing to combine the power of both types of descriptors;
- its object-based data model, providing the capability to independently describe individual objects within a scene; and
- its extensibility, provided by the DDL, which allows users to augment MPEG-7 to suit their own specific needs and the standard to keep evolving, integrating novel description tools.

This chapter presented the context that gave birth to the MPEG-7 vision, the objectives and the high-level architecture. It also listed the applications and required functionalities, along with the requirements derived from them. In addition, it presented the development process, the organization of the standard and the standardization schedule. The following chapters will describe the MPEG-7 tool set in more detail, giving the reader an idea of what the MPEG-7 toolbox looks like and how it can be used.

REFERENCES[14]

[1] *http://www.virage.com*.

[2] MPEG Home Page, *http://mpeg.telecomitalialab.com/*.

[3] MPEG Requirements Group, Introduction to MPEG-7, Doc. ISO/MPEG N4325, Sydney MPEG Meeting, July 2001, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[4] MPEG Requirements Group, MPEG-7 overview, Doc. ISO/MPEG N4317, Sydney MPEG Meeting, July 2001, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[5] MPEG Requirements Group, MPEG-7 requirements, Doc. ISO/MPEG N4320, Sydney MPEG Meeting, July 2001, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[6] MPEG Requirements Group, MPEG-7 interoperability, conformance testing and profiling, Doc. ISO/MPEG N4039, Singapore MPEG Meeting, March 2001, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[7] MPEG Requirements Group, MPEG-4 overview, Doc. ISO/MPEG N4316, Sydney MPEG Meeting, July 2001, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-4*.

[8] F. Pereira, "MPEG-4: why, what, how and when ?," *Tutorial Issue on the MPEG-4 Standard, Signal Processing: Image Communication*, **15**(4,5), pp. 271–279 (1999).

[9] L. Chiariglione, "The challenge of multimedia standardization," *IEEE Multimedia*, **4**(2), (1997).

[10] MPEG Requirements Group, MPEG-7 proposal package description (PPD), Doc. ISO/MPEG N2464, Atlantic City MPEG Meeting, October 1998.

[11] ISO/IEC 15938-6:2002, "Multimedia Content Description Interface – Part 6: Reference Software", Version 1.

[12] ISO/IEC 15938-1:2001, "Multimedia Content Description Interface – Part 1: Systems", Version 1.

[13] ISO/IEC 15938-2:2001, "Multimedia Content Description Interface – Part 2: DDL", Version 1.

[14] ISO/IEC 15938-3:2001, "Multimedia Content Description Interface – Part 3: Visual", Version 1.

[14] Although MPEG documents are in principle nonpublic documents, many of them are made public and can be accessed at the MPEG Home Page, *http://mpeg.telecomitalialab.com/*. Nonpublic MPEG documents may be obtained through the MPEG Head of Delegation of the respective country.

[15] ISO/IEC 15938-4:2001, "Multimedia Content Description Interface – Part 4: Audio", Version 1.

[16] ISO/IEC 15938-5:2001, "Multimedia Content Description Interface – Part 5: Multimedia Description Schemes", Version 1.

[17] MPEG Requirements, MPEG-7 conformance testing, Working Draft, Doc. ISO/MPEG N4038, Singapore MPEG Meeting, March 2001.

[18] ISO/IEC 15938-8:2002, "Multimedia Content Description Interface – Part 8: Extraction and Use of MPEG-7 Descriptions", Version 1.

[19] MPEG Requirements Group, MPEG-7 applications, Doc. ISO/MPEG N3934, Pisa MPEG Meeting, January 2001, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[20] MPEG Requirements Group, MPEG-7 call for proposals, Doc. ISO/MPEG N2469, Atlantic City MPEG Meeting, October 1998.

[21] MPEG Requirements Group, MPEG-7 list of proposal pre-registrations, Doc. ISO/MPEG N2567, Rome MPEG Meeting, December 1998.

[22] MPEG Requirements Group, MPEG-7 evaluation process, Doc. ISO/MPEG N2463, Atlantic City MPEG Meeting, October 1998.

[23] MPEG Requirements Group, Description of MPEG-7 content set, Doc. ISO/MPEG N2467, Atlantic City MPEG Meeting, October 1998, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[24] MPEG Requirements Group, Licensing agreement for MPEG-7 content set, Doc. ISO/MPEG N2466, Atlantic City MPEG Meeting, October 1998, *http://mpeg.telecomitalialab.com/working_documents.htm#MPEG-7*.

[25] MPEG Requirements Group, Results of MPEG-7 technology proposal evaluations and recommendations, Doc. ISO/MPEG N2730, Seoul MPEG Meeting, March 1999.

Section II
SYSTEMS

3

Systems Architecture

Olivier Avaro[1] and Philippe Salembier[2]

[1]*France Telecom, Paris, France,* [2]*Universitat Politècnica de Catalunya, Barcelona, Spain*

3.1 INTRODUCTION

The concept of 'Systems' in MPEG has evolved dramatically since the development of the MPEG-1 and MPEG-2 standards. In the past, 'Systems' referred only to overall architecture, multiplexing and synchronization. In MPEG-4, in addition to these issues, the Systems part encompasses interactive scene description, content description and programmability.

MPEG-7 brings new challenges to the Systems expertise such as languages for description representation, binary representation of descriptions and delivery of descriptions either separate or jointly with the audiovisual content. The combination of the new possibilities of describing audiovisual content offered by MPEG-7 Systems and the efficient description tools provided by the Visual, Audio and Multimedia Description Schemes (MDS) parts of the standard, promise to be the foundation of a new way of thinking about audiovisual information.

Indeed, without content description, audiovisual data is mainly an opaque series of bits. Only the decoding of these bits give some information about what the data is about and what the user can do with it. The decoding process involves, in general, complex and memory demanding operations, and requires high bandwidth in networked environments.

With the use of MPEG-7 descriptors and description schemes, MPEG-7 provides a way to get information about the audiovisual data without the need of performing the actual decoding of these data. The MPEG-7 Systems specification completes the picture by linking MPEG-7 description with the audiovisual content and providing an efficient binary representation of the description in the best MPEG tradition.

This chapter gives an overview of MPEG-7 Systems and focuses on its objectives and architecture. The first section of this chapter describes the motivations and the rationale behind the development of the MPEG-7 Systems specifications. To this goal, the requirements [4] that have guided the definition of the MPEG-7 Systems are discussed.

The second section describes the MPEG-7 Systems Architecture. A walkthrough of the MPEG-7 session highlights the different phases that a user will, in general, follow in consuming MPEG-7 descriptions. The third section of this chapter describes the notion of Access Units (AUs), which is the elementary piece of information that can be delivered. MPEG-7 is a 'toolbox' standard, providing a number of tools, sets of which are particularly suited to certain applications. This section provides a functional description of the MPEG-7 Systems tools as well. They are fully specified in [2] and [3].

3.2 OBJECTIVES

To understand the rationale behind the MPEG-7 activity, a good starting point is given by the MPEG-7 Requirements [4]. This document gives an extensive list of the objectives that needed to be satisfied by the MPEG-7 specifications. MPEG-7 Systems requirements may be categorized into two groups: traditional MPEG Systems requirements and specific MPEG-7 Systems requirements.

3.2.1 Traditional MPEG Systems Requirements

Key requirements for the development of the Systems specifications in MPEG-1, MPEG-2 and MPEG-4 were to enable the delivery of coded audio, video and user-defined private data and to incorporate timing mechanisms to facilitate synchronous decoding and presentation of these data at the client side. These requirements also constitute a part of the fundamental requirements set for MPEG-7 Systems and are further described below.

Delivery

The multimedia descriptions are to be delivered using a variety of transmission and storage protocols. Some of these delivery protocols include streaming, for example, live broadcast of the descriptions along with the content. In these cases, the multimedia descriptions have to be transmitted piece by piece, in order to match the delivery of the descriptions to clients with limited network and terminal capabilities. Delivery implies as well the definition of multiplexing tools to embed multiple MPEG-7 descriptions into a single data stream or to embed MPEG-7 description(s) into a single data stream together with associated content.

Synchronization

Typically, the different components of an audiovisual presentation are closely related in time. For some applications, the descriptions information has to be presented to the user at precise instants in time, together with the content (e.g. before, at the same time or after the content has been displayed). The MPEG-7 representation needs to allow a precise definition of the notion of delivery time so that data received in a streaming manner can be processed and presented at the right instants in time and be temporally synchronized with each other.

Stream management

Finally, the complete management of streams of audiovisual information including MPEG-7 descriptions implies the need for certain mechanisms to allow an application

to consume the content. These include mechanisms such as unambiguous location of the data, identification of the data type, description of the dependencies between data elements, association of descriptions with the content (e.g. with a content elementary stream, or part of it), access to the intellectual property information associated to the data.

3.2.2 MPEG-7 Specific Systems Requirements

In addition to these requirements, MPEG-7 brought specific needs to be solved at the Systems level: languages for the representation of description schemes and representation of binary and dynamic descriptions. The first requirement is solved by the MPEG-7 Description Definition Language (MPEG-7 DDL), which is discussed in Chapter 4 [8]. The MPEG-7 DDL is based on the Extensible Markup Language (XML) Schema language [5]. This language is used to specify the syntax or data structure of the descriptors and description schemes. The instantiation of descriptors and description schemes to describe a specific piece of audiovisual content takes the form of an XML document [6]. This format is very flexible and powerful. Moreover, there exist a large number of tools to create, process and manipulate this kind of format. However, XML documents are generally considered as verbose representations.

In some operational MPEG-7 environments, it is expected that delivery (network or storage) resources will be scarce. Therefore, data may need to be transferred in an incremental way and even in a compressed format. The incremental delivery of the description is handled by the so-called Access Units (AUs) described in Section 3.4. The compression aspects are specified in the 'Binary format for Mpeg-7' (BiM). The main requirements for the BiM are to provide a compact and streamable representation of the MPEG-7 descriptions. The BiM is equivalent[1] to the textual (XML) description defined by the MPEG-7 DDL and there is a bidirectional mapping between the BiM and the textual DDL-based representation. The textual representation is also known as TeM ('Textual format for Mpeg-7'). In addition, it is expected that some applications will use MPEG-7 BiM encoded content directly, without any intermediate step of reconstruction of the textual representation. The binary format therefore allows fast parsing of the MPEG-7 streams. It is also designed in a way that allows a BiM parser to check the syntactical correctness, for example, well formedness, and the validity regarding normative aspects of an MPEG-7 bitstream. The BiM format is described in details in Chapter 5 [7].

3.3 MPEG-7 TERMINAL ARCHITECTURE

The information representation specified by the MPEG-7 standard provides the means to describe multimedia content. The entity that makes use of such representation is generically referred to as *terminal*. This terminal may correspond to a stand-alone application or be part of an application system. The overall architecture of an MPEG-7 terminal is depicted in Figure 3.1. It highlights three layers: Application, Systems and Delivery. MPEG-7 does not specify the delivery layer nor the way the application uses the

[1] The textual and the BiM descriptions may not be perfectly equal but their XML canonical representations are equivalent. The XML canonical representation is a unique representation of a structured document that takes into account permissible changes such as order of attributes, number of white spaces and so on.

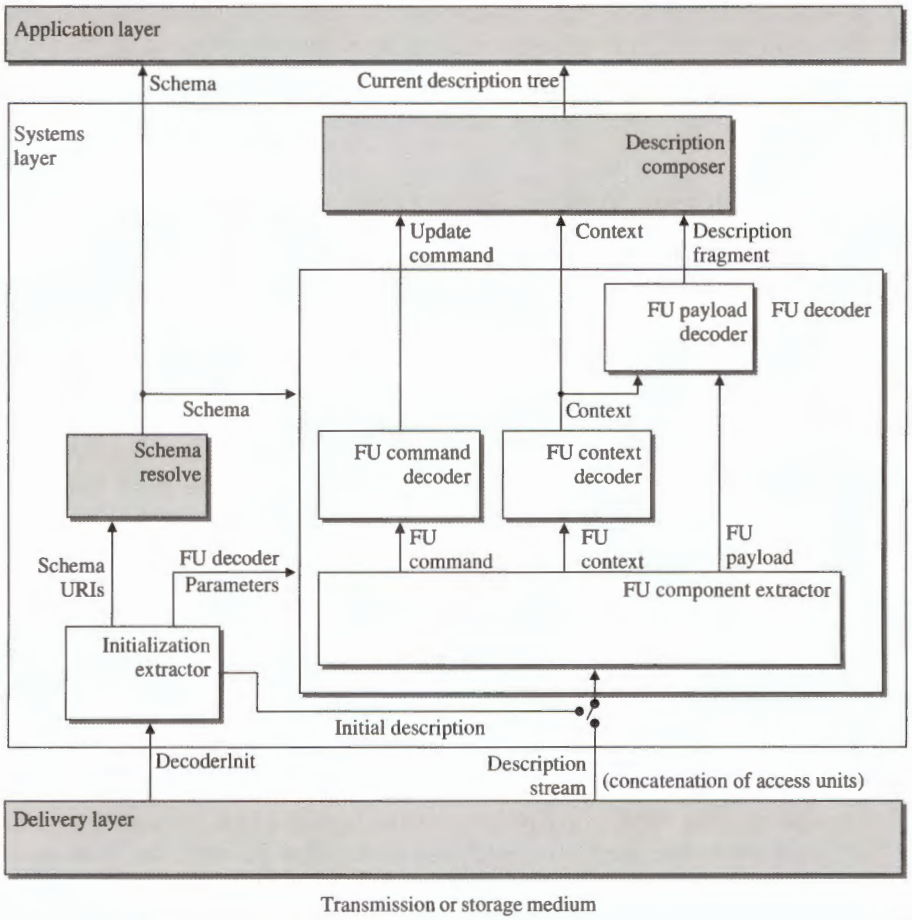


Figure 3.1 MPEG-7 systems architecture and details of the Fragment Update (FU) decoder

description. The Systems layer defines an MPEG-7 reference decoder. Note that a compliant MPEG-7 decoder does not need to implement the constituent parts as visualized in Figure 3.1, but it shall implement the normative decoding process specified in the standard.

As mentioned before, MPEG-7 description can be either in textual (XML) format or in binary format (BiM). In this chapter, we assume that the description is in textual format. Details about the case of binary description are presented in Chapter 5 [7]. Note however that the terminal architecture and the main steps of the decoding process are the same for both formats.

The transmission or storage medium corresponds to the bottom of the figure. This medium refers to the lower layers of the delivery infrastructure (network and storage layers and below). These layers deliver multiplexed streams to the Delivery layer. The transport of the MPEG-7 descriptions can occur on a variety of delivery systems. These include, for example, MPEG-2 Transport Streams, IP (Internet Protocol) or MPEG-4

(MP4) files or streams. The delivery layer encompasses mechanisms allowing synchronization, framing and multiplexing of MPEG-7 description. MPEG-7 description may be delivered independently or together with the content they describe. After the demultiplexing step, the output of the Delivery layer is a set of elementary description streams. These elementary description streams provide pieces of information about MPEG-7 description. They consist of a sequence of one or more individually accessible portions of data named AUs. An AU is the smallest data entity to which terminal-oriented timing information can be attributed. This timing information is the point in time when a specific AU becomes known to the terminal.

MPEG-7 AUs are structured as commands encapsulating the MPEG-7 description. Commands provide the dynamic aspects of the MPEG-7 description: They allow a description to be delivered in a single chunk or to be fragmented into small pieces. This feature is illustrated in Figure 3.2. The MPEG-7 description is physically a tree structure, termed the *Description Tree*. The nodes of the tree represent the pieces of information describing the content and the links between nodes represent a 'containment' relationship. The upper part of Figure 3.2 illustrates how this tree can be encapsulated in a single AU that is transmitted to the terminal. The second scenario is illustrated in the lower part of the figure. In this case, the MPEG-7 description is fragmented in three pieces that are encapsulated in different AUs. Note that MPEG-7 does not define how to fragment the description as this issue is highly application specific. The final description is reconstructed by 'adding' the content of AUs 2 and 3 to the appropriate tree node of the content of AU 1. Beside the 'add' functionality, commands also allow basic operations on the

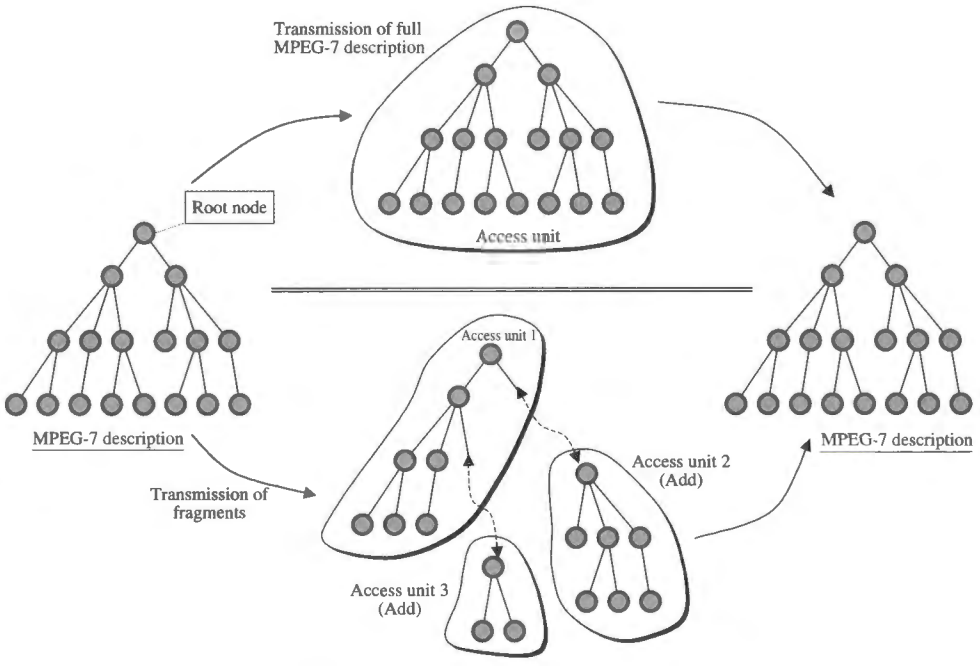


Figure 3.2 MPEG-7 description and AUs

MPEG-7 description such as updating a descriptor value, deleting part of the description or adding a new piece of information.

As can be seen in Figure 3.1, the Delivery layer provides description streams and Decoder Initialization information.

3.3.1 Decoder Initialization

The DecoderInit information is received at the Systems Layer from the Delivery Layer. It follows typically a separate processing path compared with the description stream. The DecoderInit contains a list of Schema Uniform Resource Identifiers (URIs) that identifies the schema(s) to be used to validate the description, a set of parameters to configure the decoder and an Initial Description.

The Schema URIs are processed by a schema resolver that delivers to the FU decoder the corresponding schema(s). They are used for processing and validating the description. The schema resolver is nonnormative and may, for example, retrieve schemas from a network or refer to prestored schemas. If a given Schema URI is unknown to the schema resolver, the corresponding data types in a description stream are ignored.

The Initial Description has the same general syntax and semantics as a regular AU. It goes through the FU decoder to initialize the Current Description Tree. The remaining AUs of the description stream will further update this Initial Description.

3.3.2 Processing of AUs

After the initialization of the decoder, the set of AUs can be processed. An AU is composed of an arbitrary number of FUs, each of which is extracted in sequence by the FU component extractor. Each FU consists of:

- A FU Command that specifies the type of update to be executed. There are four possible commands:
  - *addFragment* adds the fragment to the context node as the last child of the context node;
  - *replaceFragment* replaces the fragment at the context node;
  - *deleteFragment* deletes the fragment at the context node as indicated. The delete command deletes the context node and nodes that are children of the context node;
  - *reset* resets the complete description to the initial description value specified in DecoderInitialization.
- An FU Context that identifies the data type with the help of the schema and points to the location in the Current Description Tree where the FU Command applies and
- An FU Payload that provides the value, which is the piece of description, for the fragment to be added or to be replaced. This payload is passed into the FU payload decoder.

In the case of textual format, the FU payload decoder is basically a parser for the DDL [8]. By using the FU Command and FU Context, the description composer places the Description Fragment received from the FU payload decoder at the appropriate node of the Current

Description Tree (at composition time if relevant). The description composer is nonnormative because part of its task is application dependent. For example, the application may require the description composer to prune or ignore unwanted elements.

Once reconstructed, the application is then ready to exploit the MPEG-7 descriptions, possibly along with the multimedia elementary streams. The following section provide more details about the MPEG-7 AU.

3.4 ACTION UNITS

The design of the AU is illustrated in Figure 3.3. In this section, the textual version of the AU is described. The binary version will be discussed in Chapter 5 [7]. Figure 3.3 shows the structure of two elements used at the Systems level: DecoderInitialization (left) and AU (right). The notation used in Figure 3.3 is based on the Unified Modeling Language (UML) [1]. Each rectangular box corresponds to a description scheme or descriptor; paths between boxes denote composition relationships and strings such as '1,*' indicate the lower and upper bounds on the multiplicity of the relationship.

During a complete session, the first information to be received by the receiver is the DecoderInitialization. It provides the parameters necessary to decode the textual AU. As illustrated in Figure 3.3, the DecoderInitialization gives information about the Schema that is being used (through URI) and the first AU.

The AU is composed of an arbitrary number of FU Units. These FU Units are composed of FU Command, FU Context and FU Payload information. The overall strategy consists (1) in decoding the command that can be: 'add', 'delete', 'replace' or 'reset', and (2) in defining with the FU Context the location in the Current Description Tree where the command has to be applied. The location is specified by an expression relying on a subset of the XML Path Language (XPath) specification [9]. Finally, the FU Payload is the piece of description that is actually being transmitted. It encapsulates an instance of an MPEG-7 Root element (see Chapter 6 [11] for the specification of the root element).

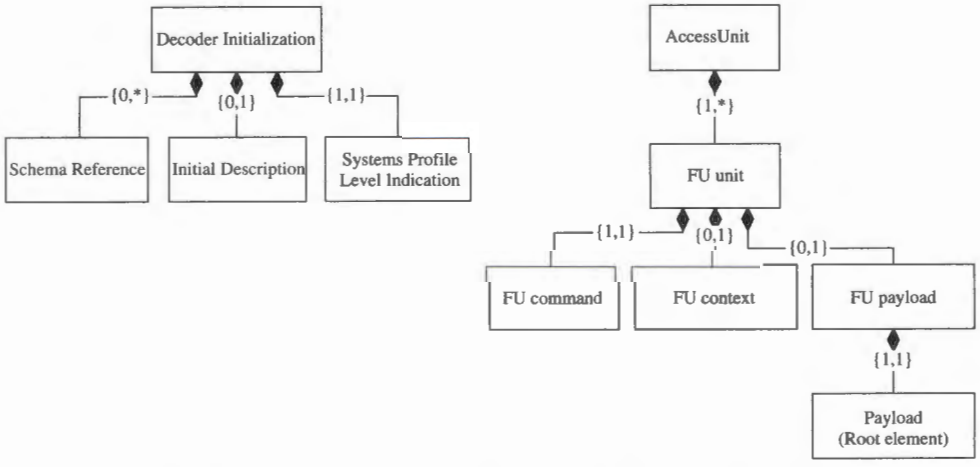


Figure 3.3 Design of the decoder initialization (left) and AU (right)

3.5 DELIVERY OF MPEG-7 DESCRIPTIONS

The delivery of MPEG-7 description on particular systems is outside the scope of the MPEG-7 standard. Existing delivery tools may be used for this purpose. MPEG is developing specifications for the transport of MPEG-7 data on MPEG-2 Systems and along MPEG-4 content. Transport of MPEG-7 description on other systems (e.g. on analog delivery systems) may be similarly developed by appropriate organizations.

The transport of MPEG-7 data on MPEG-2 Systems shall be done according to a new amendment of the MPEG-2 Systems specification [10]. This amendment provides different means to carry metadata: a synchronous transport is provided by the carriage of metadata in PES (Program Elementary Streams) packets; asynchronous transport without carousel delivery is provided by metadata sections while the Digital Storage Media-Command and Control (DSM-CC) tools can provide carousel delivery mechanisms, with or without file structures.

The transport of MPEG-7 data along MPEG-4 content is done by considering MPEG-7 data as a specific kind of MPEG-4 elementary streams. The elementary stream identification for MPEG-7 data is already provided in the MPEG-4 specification and the configuration information for such a stream is to be added to an ongoing MPEG-4 amendment. From this amendment will derive the transport of MPEG-7 data on popular delivery mechanisms already available for MPEG-4, such as IP networks and files using the MP4 file format.

3.6 CONCLUSION

The tools described above contain the majority of the functionality of MPEG-7 Systems and allow the development of compelling MPEG-7 multimedia applications. They can be summarized as follows:

- delivery, synchronization and management of multimedia descriptions using a variety of transmission and storage protocols;
- representation of dynamic multimedia descriptions through encapsulation of descriptions together with commands;
- textual and binary representation of descriptions.

The technologies considered for standardization in MPEG-7 were not all at the same level of maturity. Therefore, as for MPEG-4, MPEG-7 has organized its specification in several phases, named versions. New versions complete the current standardized toolbox with new tools and new functionality. They do not replace the tools of the previous versions. This chapter has described the MPEG-7 Systems Version 1 toolbox. Technology under consideration in MPEG-7 Systems Version 2 includes Application Program Interface (APIs) definition so that an application, possibly multiuser, can have access to the description and manipulates it according to its usage rights.

ACKNOWLEDGMENT

The MPEG-7 Systems specification reflects the results of teamwork within a worldwide project in which many people invested time and energy. The authors would like to thank them all and hope that the paper reflects accurately the understanding of the group.

REFERENCES[2]

[1] S. S. Alhir, *UML in a Nutshell*, O'Reilly & Associates, Sebastopol, Calif., 1998.

[2] ISO/IEC 15938-1:2001, "Multimedia content description interface – Part 1 Systems", Version 1.

[3] ISO/IEC 15938-2:2001, "Multimedia content description interface – Part 2 Description definition language", Version 1.

[4] MPEG Requirements Group, "MPEG-7 Requirements", Doc. ISO/MPEG N4320, Sydney MPEG Meeting, July 2001.

[5] XML Schema Part 0: Primer, W3C Candidate Recommendation, 2 May 2001. *http://www.w3.org/TR/xmlschema-0/*.

[6] XML: Extensible Markup Language 1.0 (Second Edition), W3C Recommendation, 6 October 2000. *http://www.w3.org/TR/REC-xml/*.

[7] J. Heuer, C. Thienot, M. Wollborn, Binary Format, Chapter 5.

[8] J. Hunter and C. Seyrat, Description Definition Language, Chapter 4.

[9] XML Path Language (Xpath) Version 1.0, W3C Recommendation, 16 November 1999. *http://www.w3.org/TR/xpath.html*.

[10] "ISO/IEC 13818-1:20 00/FPDAM1", Version 1.

[11] P. Salembier, J. Smith, Overview of Multimedia Description Schemes and Schema Tools, Chapter 6.

[2] Although MPEG documents are in principle nonpublic documents, many of them are made public and can be accessed at the MPEG Home Page, *http://mpeg.telecomitalialab.com/*. Nonpublic MPEG documents may be obtained through the MPEG Head of Delegation of the respective country.

4

Description Definition Language

Jane Hunter[1] and Claude Seyrat[2]

[1]*DSTC Pty, Ltd., Australia,* [2]*Expway, Paris, France*

4.1 INTRODUCTION

The DDL provides the foundations for the MPEG-7 standard. It provides the language for defining the structure and content of MPEG-7 documents. The DDL is not a modeling language such as Unified Modeling Language (UML) but a *schema* language to represent the results of modeling audiovisual data, (i.e. descriptors and description schemes) as a set of syntactic, structural and value constraints to which valid MPEG-7 descriptors, description schemes, and descriptions must conform. It also provides the syntactic rules by which users can combine, extend and refine existing description schemes and descriptors to create application-specific description definitions or schemas. The purpose of a schema is to define a class of XML (Extensible Markup Language) documents. The purpose of an MPEG-7 schema is to define a class of MPEG-7 documents. MPEG-7 *instances* are XML documents that conform to a particular MPEG-7 schema (expressed in the DDL) and that describe audiovisual content.

According to the MPEG-7 DDL requirements [1], the DDL must be capable of expressing structural, inheritance, spatial, temporal, spatiotemporal and conceptual relationships between the elements within a description scheme and between description schemes. It must provide a rich model for links and references between one or more descriptions and the data that it describes. It must be platform and application independent, machine-readable and preferably human-readable. It must be capable of specifying descriptor data types, both primitive (integer, text, date, time) and composite (histograms, enumerated types).

In addition, a DDL Parser is required that is capable of validating the syntax of MPEG-7 schemas, which comprises description schemes (content and structure) and descriptors (data types). Given an MPEG-7 description (encoded in XML), the parser must also be able to check its conformance to the rules expressed in the corresponding MPEG-7 schema (a set of description schemes and descriptors defined using the DDL).

At the 51st MPEG meeting in Noordwijkerhout in March 2000, it was decided to adopt the W3C's XML Schema Language as the MPEG-7 DDL. However, because XML

Schema Language has not been designed specifically for audiovisual content, certain extensions are necessary in order to satisfy all of the MPEG-7 DDL requirements. Hence the DDL consists of the following logical components, which are described in Sections 4.3, 4.4 and 4.5: XML Schema structural components, XML Schema data types and MPEG-7-specific extensions.

In the remainder of this chapter, we will describe the events that have led to the current DDL and its key components and features. Complete specifications of the DDL can be found in the MPEG-7 DDL Final Draft International Standard (FDIS) [2] and the W3C XML Schema Recommendations [3–5].

4.2 HISTORICAL BACKGROUND

In response to the MPEG-7 Call for Proposals in October 1998, the MPEG-7 DDL evaluation team compared and evaluated ten DDL proposals at the MPEG-7 Test and Evaluation Meeting in Lancaster in February 1999. A summary report [6] was produced that concluded that XML [7] should be used as the syntax for the MPEG-7 DDL. There was also a consensus that the MPEG-7 DDL must support the validation of structural, relational and data typing constraints as well as the expression of semantics through the addition of richer constraints such as inheritance.

Although none of the proposals could satisfy all of the requirements, it was decided to base the DDL on Distributed Systems Technology Centre (DSTC's) proposal, P547 [8], with the integration of ideas and components from other proposals and contributors. In addition, the strategy was to continue monitoring and liaising with related efforts in the W3C community, in particular the XML Schema [9], XML Linking Language (XLink) [10] and XML Path Language (XPath) [11] Working Groups (WGs).

In May 1999, the XML Schema WG produced the first version of a two-part working draft of the XML Schema Language: XML Schema Part 1: Structures [4] and XML Schema Part 2: Data types [5]. Preliminary encoding of the Multimedia Description Schemes (MDSs) [12] using XML Schema Language demonstrated its suitability as a basis for the DDL. However, reservations were raised at the 48th MPEG meeting in Vancouver in July 1999 concerning MPEG–7's dependency on the output and time schedule of W3C XML Schema WG. As a result, the decision was made to develop a proprietary MPEG-7-specific language in parallel with the XML Schema Language developments within W3C. A new grammar based on DSTC's proposal but using MPEG-7 terminology (description schemes and descriptors) and with modifications to ensure simple mapping to XML Schema, was developed.

However, when XML Schema moved to the Last Call for Review stage in March 2000, it was decided (at the 51st MPEG meeting in Noordwijkerhout in March 2000) to adopt XML Schema Language, with additional MPEG-7-specific extensions, as the DDL. This decision was made in recognition of XML Schema's growing stability and expected widespread adoption, the availability of open source XML Schema parsers and the release of Last Call for Review Working Drafts by the W3C.

A detailed evaluation of XML Schema revealed that although XML Schema satisfies the majority of the MPEG-7 DDL requirements, there are certain requirements that are not satisfied and some features that are problematic. A list of problem issues and

feature requests was submitted to the XML Schema WG in response to the Last Call for Review [13]. Certain high priority features, which are not defined in XML Schema, have been implemented as MPEG-7-specific extensions. Validation of these extensions will be through extensions to existing XML Schema parsers implemented during the development of MPEG-7 parsers.

4.3 XML SCHEMA STRUCTURAL COMPONENTS

The purpose of this section is to describe the language constructs provided by XML Schema, which can be used to constrain the content structure and attributes associated with MPEG-7 description schemes and descriptors.

XML Schema consists of three categories of schema components. The primary components are

- Namespaces and the schema wrapper around the definitions and declarations;
- Element declarations;
- Attribute declarations;
- Type definitions: simple, complex, derived and anonymous.

The secondary components are

- Attribute group definitions;
- Model group definitions;
- Identity-constraint definitions;
- Notation declarations.

The third group is the 'helper' components, which contribute to the other components and cannot stand alone:

- Annotations;
- Model groups;
- Particles;
- Wildcards.

In this chapter, we only describe those features of most importance to MPEG-7 – the primary components and group definitions. Details of other components not described here can be found in the XML Schema Recommendations [3–5].

4.3.1 Namespaces and the Schema Wrapper

XML namespaces [14] provide a simple method for assigning universally unique names to element types or attribute names within a XML document, so that they can be reused in other XML documents. According to [14], an XML *namespace* is a collection of names, identified by a Uniform Resource Identifier (URI) reference, which are used in XML documents as element types and attribute names. *Qualified* names consist of a namespace prefix (which maps to a URI reference), a colon separator and the local part (an element type or attribute name). This combination produces identifiers for schema components, which are universally unique and thus reusable, for example, *mpeg7:VideoSegmentType*.

In the context of MPEG-7, the namespace mechanism enables descriptors and description schemes from multiple different MPEG-7 schemas to be reused and combined to create new schemas. Every schema definition must begin with a preamble in order to identify the current namespace and other imported namespaces. The mandatory preamble consists of an XML element 'schema', which includes the following attributes:

xmlns: A URI to the XML Schema namespace;
xmlns:mpeg7: A URI to the MPEG7 schema to be used for validating MPEG-7 description schemes and descriptors;
targetNamespace: The URI by which the current schema is to be identified;
xmlns:xxx: References to other imported schemas and abbreviations for referring to definitions in these external schemas, for example, *xmlns:dc* in the example below associates the prefix *dc:* with the Dublin Core namespace, which is located at the given URI.

```
<schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:mpeg7="http://www.mpeg7.org/2001/MPEG-7_Schema"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  targetNamespace="http://www.dstc.org/2001/DSTC_MPEG-7_Schema">
  ...
</schema>
```

4.3.2 Element Declarations

Element declarations enable the appearance in document instances of elements with specific names and types. An element declaration specifies a type definition for a schema element either explicitly or by reference, and may provide occurrence (through the *minOccurs* and *maxOccurs* attributes) and default information (through the *default* attribute). For example, the element declaration below associates the name *Country* with an existing type definition, *countryCode*, specifies that the default value for the Country element is 'en' (England) and that the *Country* element can occur zero or more times.

```
<element      name="Country"    type="countryCode"    default="en"
          minOccurs="0" maxOccurs="unbounded"/>
```

The values for minOccurs and maxOccurs are computed in the following way:

minOccurs=

- the actual value of the minOccurs attribute, if present,
- otherwise 1.

maxOccurs=

- unbounded, if the maxOccurs attribute equals unbounded;
- otherwise the actual value of the maxOccurs attribute, if present;
- otherwise 1.

Sometimes it is preferable to reference an existing element rather than declare a new element.

```
<element ref="Country" minOccurs="1"/>
```

This declaration references an existing element (*Country)* that was declared elsewhere in the schema. The value of the ref attribute must reference a global element, that is, one that has been declared under *schema* rather than as part of a complex type definition. The consequence of this declaration is that an element called *Country* must appear at least once in an instance document and its content must be consistent with that element's type, the *countryCode*.

4.3.3 Attribute Declarations

Attribute declarations enable the appearance in document instances of attributes with specific names and types by associating an attribute name with a simple data type. Within an attribute declaration, the *use* attribute specifies presence (*required|optional|prohibited*). The default value of *use* is *optional*. The declared attribute can have a fixed or a default value specified by the *default* or *fixed* attribute. The declaration below indicates that the appearance of a *lang* attribute (in the element *Annotation*) is optional and its default value is 'en-uk'.

```
<attribute name="lang" type="language" use="optional" default="en-uk"/>

<element name="Annotation"/>
   <complexType/>
      <attribute ref="lang"/>
   </complexType>
</element>
```

Below is a valid instance of an element of the above type:

```
<Annotation lang="en-us"/>
```

4.3.4 Type Definitions

In XML Schema there is a fundamental distinction between type definitions (which create new types) and declarations, which enable the appearance in document instances of elements and attributes with specific names and types.

Type definitions define internal schema components, which can be used in other schema components such as element or attribute declarations or other type definitions. For example, below we first define the (simple) type *PostCode* that is a string of length 7:

```
<simpleType name="Postcode"/>
   <restriction base="string">
      <length value="7"/>
   </restriction>
</simpleType>
```

We can then declare elements or attributes, which are of this type and which are to appear in instantiated XML documents. For example, we can declare an element MyPostcode, which is of type Postcode, which is to appear in an instance document:

```
<element name="MyPostcode" type="Postcode"/>
```

XML Schema provides *simple*, *complex* and *derived* type definitions, as described below.

Simple type definitions

Simple types cannot have children elements and cannot carry attributes. Both elements and attributes can be declared to have simple types. XML Schema provides a large number of simple types through a set of built-in primitive types and a set of built-in derived types (derived from the primitive types) [5]. These built-in data types are described in Section 4.4. In addition to the built-in simple types, new simple types can be derived by the application of restrictions to other simple types. These restrictions are specified by facets such as the *enumeration* facet on a string or the *minInclusive* and *maxInclusive* facets on an integer, as shown in the examples below. The facets that are applicable to each data type are listed in Appendix C of [5].

```
<simpleType name="directionType">
   <restriction base="string">
      <enumeration value="left"/>
      <enumeration value="right"/>
   </restriction>
</simpleType>
<simpleType name="unsigned6">
   <restriction base="nonNegativeInteger">
      <minInclusive value="0"/>
      <maxInclusive value="63"/>
   </restriction>
</simpleType>

<element name="direction" type="directionType"/>
<element name="length" type="unsigned6"/>
```

Below are valid instances of the defined elements:

```
<direction>left</direction>
<length>36</length>
```

In addition to atomic (indivisible) simple types, XML Schema also provides two aggregate simple types: *list* types and *union* types. List types are composed of sequences of atomic types. Union types enable element or attribute values to be instances of a type drawn from the union of multiple atomic and list types. These two aggregate types are described in Sections 4.4 and 4.5.

Complex type definitions

Unlike simple types, complex types allow children elements in their content and may carry attributes. Complex type definitions provide

- Constraints on the appearance and nature of attributes;
- Constraints on the appearance and nature of children elements;
- Derivation of complex types from other simple or complex types through extension or restriction.



The set of the rules that describe the content of an element is called a content model. It is possible to constrain the nature of the content model of a complexType to the following:

- *empty* – no child elements only attributes;
- *mixed* – character data appears between elements and their children. The *mixed* attribute must be set to 'true';
- *complexContent* – the default content type which consists of elements and attributes. Three compositors are provided to structure unnamed groups of elements within complexContent:
  - *sequence* – the elements may appear in the same order in which they are declared;
  - *choice* – only one of the elements may appear in an instance;
  - *all* – all of the elements may appear once or not at all and in any order.
- *simpleContent* – use when deriving a complexType from a simpleType because we want to add an attribute to a simpleType. This indicates that the new type contains only character data (i.e. does not contain any elements) but may also have attributes associated with it.

New complex types are defined using the *complexType* element and such definitions typically contain a set of element declarations, element references and attribute declarations.

Below is an example of a complex type with a complex content model and a valid instance of this example:

```
<complexType name="OrganizationType">
   <sequence>
      <element name="OrgName"type="string"/>
      <element name="ContactPerson" type="IndividualType"
                                 minOccurs="1" maxOccurs="unbounded"/>
      <element name="Address" type="PlaceType"
                                 minOccurs="1" maxOccurs="1"/>
   </sequence>
   <attribute name="id" type="ID" use="required"/>
</complexType>

<element name="ProdComp" type="OrganizationType"/>
```

Below is a valid instance corresponding to the above complexType definition:

```
<ProdComp id="ORG754">
  <OrgName>DSTC Pty Ltd</OrgName>
  <ContactPerson>Liz Armstrong</ContactPerson>
  <Address>University of Qld</Address>
</ProdComp>
```

The consequence of this definition is that any *ProdComp* elements appearing in an MPEG-7 description must consist of an *OrgName* element, one or more *ContactPerson* elements and one *Address* element. The first of these elements will contain a string, the second will contain the complexType *IndividualType* and the third will contain the complexType *PlaceType*. Finally, any element whose type is declared to be *Organization* must also appear with an attribute called *id*, which must contain an *ID* (unique identifier).

To define a type whose content is empty, we define a type that allows only elements in its content, but we do not actually declare any elements, only attributes. Below is an example of the *empty* content model and a valid instance of this example:

```
<element name="Price">
   <complexType>
      <attribute name="currency" type="currencyCode"/>
      <attribute name="value" type="decimal"/>
   </complexType>
</element>

<Price currency="EU" value="423.46"/>
```

Below is an example of the *mixed* content model and a valid instance of the example:

```
<element name="Introduction">
   <complexType mixed="true">
      <sequence>
         <element name="Name" type="string"/>
      </sequence>
   </complexType>
</element>

<Introduction> Dear Ms. <Name> Hetty Wilson </Name> , </Introduction>
```

Derived types

It is possible to derive new complexTypes by *extension* or *restriction* of simple or complex base type definitions.

A complex type extends another by having additional content model particles at the end of the other definition's content model, or by having additional attribute declarations or both. A new complexType can be derived by extending a simpleType through the addition of attributes. To indicate that the content model of the new type contains only character data with attributes and no elements, the *simpleContent* element is used. In the following example, the *source* and *target* attributes are added to the simple string base type to define the *RelationType*.

```
<complexType name="RelationType">
   <simpleContent>
      <extension base="string">
         <attribute name="source" type="IDREF" use="optional"/>
         <attribute name="target" type="IDREF" use="optional"/>
      </extension>
   </simpleContent>
</complexType>

<element name="Relation" type="RelationType"/>

<Relation source="edge1" target="edge2"> morph </Relation>
```

A new complexType may also be derived by extending an existing complexType. In the example below, the *PersonName* type is extended through the addition of a new *Nickname* element, to create the *FriendName* type. The *complexContent* element is required

to indicate that we intend to restrict or extend the content model of a complex type and that the new type contains only elements and no character data.

```
<complexType name="PersonNameType">
   <sequence>
      <element name="Title" type="string" minOccurs="0"/>
      <element name="Forename" type="string"
                                 minOccurs="0" maxOccurs="unbounded"/>
      <element name="Surname" type="string"
                                 minOccurs="1" maxOccurs="1"/>

   </sequence>
</complexType>

<element name="PersonName" type="PersonNameType"/>

<complexType name="FriendNameType">
   <complexContent>
      <extension base="PersonNameType">
         <sequence>
            <element name="Nickname" type="string"/>
         </sequence>
      </extension>
   </complexContent>
</complexType>

<element name="FriendName" type="FriendNameType"/>
```

Below are valid instances of the elements declared above:

```
<PersonName>
   <Title>Prof.</Title>
   <Forename>Simon</Forename>
   <Forename>Daniel</Forename>
   <Forename>Benjamin</Forename>
   <Surname>Kaplan</SurName>
</PersonName>

<FriendName>
   <Title>Dr.</Title>
   <Forename>Jocelyn</Forename>
   <Surname>Richards</Surname>
   <Nickname>Jos</Nickname>
</FriendName>
```

A type definition whose declarations or facets are in a one-to-one relation with those of another specified type definition, with each in turn restricting the possibilities of the one it corresponds to, is said to be a restriction. The specific restrictions might include narrowed ranges or reduced alternatives. Members of a type, A, whose definition is a restriction of the definition of another type, B, are always members of type B as well.

In the example below, the *SimpleName* type is derived through restriction of the *PersonName* type. The restriction is on the occurrence constraints of the *Title* and *Forename* elements that are no longer unbounded but restricted to be equal to 1.

```
<complexType name="SimpleNameType">
  <complexContent>
```

```
    <restriction base="PersonNameType">
      <sequence>
        <element name="Title" minOccurs="1" maxOccurs="1"/>
        <element name="Forename" minOccurs="1" maxOccurs="1"/>
        <element name="Surname" minOccurs="1" maxOccurs="1"/>
    </sequence>
    </restriction>
  </complexContent>
</complexType>
<element name="SimpleName" type="SimpleNameType"/>
```

Below is a valid instance of an element of the above type because it has precisely one Title and one Forename element:

```
<SimpleName>
   <Title>Prof.</Title>
   <Forename>Simon</Forename>
   <Surname>Kaplan</Surname>
</SimpleName>
```

XML Schema allows derived types to appear in a description instead of expected types. In the preceding example, a description could use the *FriendNameType* everywhere a *PersonNameType* is expected. In this case, the derived type must be identified in the instance using the *xsi:type* attribute.

```
<PersonName xsi:type="FriendNameType">
   <Title>Dr.</Title>
   <Forename>Jocelyn</Forename>
   <Surname>Richards</Surname>
   <Nickname>Jos</Nickname>
</PersonName>
```

Anonymous type definitions

Schemas can be constructed by defining named types and then declaring elements that reference the types using the '*element name=.. type=..*' construction as illustrated in the examples shown above. This style of schema construction is straightforward but it can become unwieldy if many of the defined types are only referenced once and contain few constraints. In these cases, a type can be more succinctly defined as an anonymous type, which saves the overhead of having to be named and explicitly referenced. For example, the element below contains a complexType definition that is unnamed or anonymous:

```
<element name="Affiliation">
  <complexType>
    <choice>
      <element name="Organization" type="OrganizationType"/>
      <element name="PersonGroup" type="PersonGroupType"/>
    </choice>
  </complexType>
</element>
```

4.3.5 Group Definitions

The *attributeGroup* and *group* elements provide mechanisms for creating and naming groups of attributes and groups of elements respectively. Such groups can then be incorporated by reference into complexType definitions.

```
<attributeGroup name="person_attributes">
   <attribute name="weight" type="decimal"/>
   <attribute name="height" type="decimal"/>
   <attribute name="age" type="integer"/>
</attributeGroup>

<complexType name="Person">
   <sequence>
      <element ref="Name"/>
   </sequence>
   <attributeGroup ref="person_attributes"/>
</complexType>
```

Three compositors (*sequence*, *choice* and *all*) are also provided to construct unnamed groups of elements within complex content. These are described in the previous section on ComplexType Definitions.

In the example below, the *ContactGroup* is defined as a choice between the two elements, *Organization* and *Person*. The *PublisherType* is then defined as a sequence of *ContactGroup* and *Address* with an *id* attribute.

```
<group name="ContactGroup">
   <choice>
      <element ref="Organization"/>
      <element ref="Person"/>
   </choice>
</group>

<complexType name="PublisherType">
   <sequence>
      <group ref name="ContactGroup"/>
      <element ref="Address"/>
   </sequence>
   <attribute name="id" type="ID" use="optional"/>
</complexType>
```

Example of a valid instance:

```
<Publisher id="ORG123">
  <Organization>DSTC Pty Ltd<Organization/>
  <Address>Uni. Of Qld</Address>
</Publisher>
```

4.4 XML SCHEMA DATA TYPES

This section describes the built-in primitive data types, the built-in derived data types and mechanisms for defining customized derived data types such as facets, lists and union data

types. These facilities can be used to constrain the possible values of MPEG-7 descriptors within instantiated descriptions.

4.4.1 Built-in Primitive Data Types

The following built-in primitive data types are provided within XML Schema Data types. Precise details including the lexical and canonical representations and the constraining facets for each of these data types can be found in [5]:

string: character strings;
boolean: {true/false};
decimal: finite-length sequence of decimal digits separated by a period as a decimal indicator;
float: IEEE single-precision 32-bit floating-point type. Floats have a lexical representation of a mantissa (decimal number) followed by the character 'E' or 'e' followed by an exponent (integer), for example, 12.78E-2;
double: IEEE double-precision 64-bit floating-point type. Doubles have a lexical representation of a mantissa (decimal number) followed by the character 'E' or 'e' followed by an exponent (integer), for example, 12.78E-2;
duration: a duration of time represented in the [ISO8601] [15] format P*n*Y*n*DT*n*H*n*M*n*S, where *n*Y represents the number of years, *n*M the number of months, *n*D the number of days, 'T' is the date or time separator, *n*H the number of hours, *n*M the number of minutes and *n*S the number of seconds. For example, to indicate a duration of 1 year, 2 months, 3 days, 10 hours and 30 minutes, one would write: P1Y2M3DT10H30M;
dateTime: a specific instant of time represented in the [ISO 8601] [15] extended format CCYY-MM-DDThh:mm:ss where 'CC' represents the century, 'YY' the year, 'MM' the month and 'DD' the day, 'T' is the date or time separator and 'hh', 'mm', 'ss' represent hour, minute and second, respectively;
time: an instant of time that recurs everyday, represented using the format: hh:mm:ss.sss with optional following time zone (TZ) indicator. For example, to indicate 1:20 P.M. for Eastern Standard Time, which is five hours behind Coordinated Universal Time (UTC), one would write: 13:20:00–05:00;
date: a calendar date, represented using the CCYY-MM-DD. For example, to indicate May the 31st, 1999, one would write: 1999-05-31;
gYearMonth: a specific Gregorian month in a specific Gregorian year, represented using the format CCYY-MM;
gYear: a Gregorian calendar year, specified using the format CCYY;
gMonthDay: a Gregorian date that recurs, specifically a day of the year such as the third of May, and which is represented using the format – MM-DD;
gDay: a Gregorian day that recurs, specifically a day of the month such as the 5th of the month. It is represented using the format: – DD;
gMonth: a Gregorian month that recurs every year. It is represented using the format: – MM–;
base64Binary: base64-encoded arbitrary binary data represented by finite-length sequences of binary octets;
hexBinary: arbitrary hex-encoded binary data represented by finite-length sequences of binary octets in which each binary octet is encoded as a character tuple, consisting

of two hexadecimal digits ([0-9a-fA-F]) representing the octet code. For example, '0FB7' is a hex encoding for the 16-bit integer 4023 (whose binary representation is 111110110111);

anyURI: a Uniform Resource Identifier Reference (URI), which can be absolute or relative and may have an optional fragment [16];

Qname: represents XML qualified names that consist of tuples {namespace name, local part}, where namespace name is an any URI and local part is an NCName;

NOTATION: the NOTATION attribute type from [XML 1.0 [7]], which is the set of all names of notations declared in the current schema. NOTATION should not be used directly in a schema. Only data types that are ·derived· from NOTATION by specifying a value for ·enumeration· can be used in a schema and NOTATION should be used only on attributes.

4.4.2 Built-in Derived Data Types

A number of built-in derived data types (derived by applying facets to the primitive data types) are provided within XML Schema Data types. The complete definitions for the built-in derived data types, which are listed below, can be found in [5]:

normalizedString: white space normalized strings that do not contain the carriage return (#xD), line feed (#xA) nor tab (#x9) characters;

token: tokenized strings that do not contain the line feed (#xA) nor tab (#x9) characters, that have no leading or trailing spaces (#x20) and that have no internal sequences of two or more spaces;

language – the set of all strings that are valid language identifiers as defined in RFC, 1766 [17];

NMTOKEN: the NMTOKEN attribute type from XML 1.0 (Second Edition) [7] – the set of tokens that match the Nmtoken production shown in Reference 7;

NMTOKENS: the NMTOKENS attribute type from Reference 7 – the set of finite, nonzero-length sequences of NMTOKENs;

Name – the set of all strings that match the Name production of XML 1.0 [7];

NCName: XML 'non-colonized' Names – the set of all strings that match the NCName production of Namespaces in XML [14];

ID: the ID attribute type from XML 1.0 [7];

IDREF, IDREFS: the IDREF and IDREFS attribute types from [XML 1.0] [7].;

ENTITY, ENTITIES: the ENTITY and ENTITIES attribute types from XML 1.0 [7];

integer: a finite-length sequence of decimal digits with an optional leading sign. If the sign is omitted, '+' is assumed. For example: -1, 0, 12 678 967 543 233, $+100\,000$. The ·value space· of integer is the infinite set $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$;

nonPositiveInteger: derived from integer by setting maxInclusive to 0, the value space is the infinite set $\{\ldots, -2, -1, 0\}$;

negativeInteger: derived from integer by setting maxInclusive to -1, the value space is the infinite set $\{\ldots, -2, -1\}$;

nonNegativeInteger: derived from integer by setting minInclusive to 0, the value space is the infinite set $\{0, 1, 2, \ldots\}$;

positiveInteger: derived from integer by setting minInclusive to 1, the value space is the infinite set $\{1, 2, \ldots\}$;

long: derived from integer by setting the value of maxInclusive to be 9 223 372 036 854 775 807 and minInclusive to be – 9 223 372 036 854 775 808;
int: derived from long by setting the value of maxInclusive to be 2 147 483 647 and minInclusive to be – 2 147 483 648;
short: derived from int by setting the value of maxInclusive to be 32 767 and minInclusive to be – 32 768;
byte: derived from short by setting the value of maxInclusive to be 127 and minInclusive to be −128;
unsignedLong: derived from nonNegativeInteger by setting the value of maxInclusive to be 18 446 744 073 709 551 615;
unsignedInt: derived from unsignedLong by setting the value of maxInclusive to be 4 294 967 295;
unsignedShort: derived from unsignedInt by setting the value of maxInclusive to be 65 535;
unsignedByte: derived from unsignedShort by setting the value of maxInclusive to be 255.

4.4.3 Facets

A derived data type is defined by applying constraining facets to a primitive data type or another derived data type. The table below lists the facets, which are provided to generate customized data types.

Bounds facets	minInclusive, minExclusive maxInclusive, maxExclusive.
Numeric facets	totalDigits, fractionDigits
Pattern facet	Pattern
Enumeration facet	Enumeration
Length facets	Length, minLength, maxLength
White space facet	White space

The example below illustrates the application of the *minInclusive* and *maxInclusive* facets to a *float* data type to restrict elements of type *height* to between 0.0 and 120.0:

```
<simpleType name="height">
   <restriction base="float">
      <minInclusive value="0.0"/>
      <maxInclusive value="120.0">
   </restriction>
</simpleType>
```

The example below illustrates the use of the *pattern* facet. The data type is constrained to strings that match a specific pattern defined using a regular expression. The following example restricts elements of type *PhoneNum* to strings of three digits followed by a dash followed by four digits. The precise syntax of regular expressions can be found in Appendix F of [5]:

```
<simpleType name="PhoneNum">
   <restriction base="string">
```

```
      <pattern value="3-4"/>
   </restriction>
</simpleType>
```

The example below illustrates the application of the *enumeration* facet to a string to restrict the possible values of 'temporal_relation' to: *before*, *during*, *after* or *contains*. The enumeration facet can be applied to almost every simple type (except the boolean type) to limit the type to a set of distinct values:

```
<simpleType name="temporal_relation"/>
   <restriction base="string">
      <enumeration value="before"/>
      <enumeration value="during"/>
      <enumeration value="after"/>
      <enumeration value="contains"/>
   </restriction>
</simpleType>
```

4.4.4 The List Data Type

List types are composed of sequences of atomic types, separated by white space. XML Schema has three built-in list types, NMTOKENS, IDREFS and ENTITIES. In addition, new list types can be created by derivation from existing atomic types. List types cannot be created from existing list types or from complex types but facets (*length, minLength, maxLength* and *enumeration*) can be applied to derive new list types.

Examples of list type definitions and a valid instance:

```
<simpleType name="integerVector">
   <list itemType="integer"/>
</simpleType>

<simpleType name="integerVector4">
   <restriction base="integerVector">
      <length value="4"/>
   </restriction>
</simpleType>

<integerVector4>5 8 11 2</integerVector4>
```

4.4.5 The Union Data Type

Union types enable element or attribute values to be one or more instances of one type drawn from the union of multiple atomic and list types. In the example below, the element Unsigned6OrDirection can have a value that is of either type Unsigned6 or Direction, because it is defined as the union of these two types.

```
<element name="Unsigned6OrDirection">
  <simpleType>
    <union memberTypes="unsigned6 directionType"/>
  </simpleType>
</element>
```

The examples below are both valid instances of this element:

```
<Unsigned6OrDirection> 45 </Unsigned6OrDirection>
<Unsigned6OrDirection> left </Unsigned6OrDirection>
```

4.5 MPEG-7-SPECIFIC EXTENSIONS

In order to satisfy the MPEG-7 DDL requirements, it has been necessary to add the following features to XML Schema:

- Array and matrix data types;
- Built-in derived data types *basicTimePoint* and *basicDuration*.

4.5.1 Array and Matrix Data Types

MPEG-7 requires DDL mechanisms to:

- restrict the size of multidimensional matrices to a predefined facet value in a schema definition;
- restrict the size of one-dimensional arrays or multidimensional matrices to an attribute at time of instantiation.

Using the *list* data type, two methods are provided for specifying sizes of (1D) arrays and multidimensional matrices. A new *mpeg7:dimension* facet, which is a list of positive integers, is provided to enable the specification of the dimensions of a fixed-size matrix. Because the *dimension* facet is an MPEG-7 extension that is not compliant with XML Schema Language, it needs to be wrapped inside the `<annotation><appinfo>` to ensure XML Schema parsers ignore it and MPEG-7 parsers will validate (and possibly process) it.

The example below illustrates the definition and instantiation of an integer matrix with 3 rows and 4 columns:

```
<simpleType name="IntMatrix2D">
   <list itemType="integer">
      <annotation><appinfo>
         <mpeg7:dimension value="unbounded unbounded"/>
      </appinfo></annotation>
   </list>
</simpleType>

<simpleType name="IntMatrix3x4">
   <restriction base="IntMatrix2D">
      <annotation><appinfo>
         <mpeg7:dimension value="3 4"/>
      </appinfo></annotation>
   </restriction>
</simpleType>

<element name="IntMat3x4" type="IntMatrix3x4"/>

<IntMat3x4>
   5 8 9 4
   7 6 1 2
   1 3 5 8
</IntMat3x4>
```

The special *mpeg7:dim* attribute is also provided to support parameterized array and matrix sizes. It specifies the dimensions to be applied to a list type at the time of instantiation and is defined in the mpeg-7 namespace as a list of positive integers.

```
<simpleType name="dim">
   <list itemType="integer"/>
</simpleType>

<complexType name="NDimIntegerArray">
   <simpleContent>
      <extension base="listOfInteger">
         <attribute ref="mpeg7:dim" />
      </extension>
   </simpleContent>
</complexType>

<element name="IntegerMatrix" type="NDimIntegerArray" />
```

In the following example, a matrix with 2 rows and 4 columns is specified at the time of instantiation using *mpeg7:dim*.

```
<IntegerMatrix mpeg7:dim="2 4">
  1 2 3 4
  5 6 7 8
</IntegerMatrix>
```

4.5.2 Built-in Derived Data Types

In addition to the built-in derived types provided by XML Schema Datatypes, the following built-in data types are also provided by MPEG-7 to explicitly satisfy the requirements of MPEG-7 implementers:

The *basicTimePoint* data type, specifies a time point according to the Gregorian dates, day time and the TZ. The format is based on the ISO 8601 [15] standard. To reduce conversion problems only a subset of the ISO 8601 formats is used [2].

The *basicDuration* data type specifies the duration of a time period according to days and time of day. The format is based on the ISO 8601 [15] standard. To reduce conversion problems, only a subset of the ISO 8601 formats is used [2]. Fractions of a second are specified according to the *basicTimePoint* datatype.

4.6 CONCLUSION

This chapter has provided an overview of the MPEG-7 DDL. XML Schema has been chosen as the basis for the DDL, because of its widespread adoption as a schema language for constraining the structure and content of XML documents, its ability to satisfy the MPEG-7 DDL requirements and the ready availability of XML Schema tools and parsers. Consequently this chapter has primarily provided an overview of the XML Schema Language. In time we hope that the MPEG-7-specific extensions to XML Schema will be incorporated as part of a Type library provided within future versions of XML Schema – this will make the MPEG-7 DDL fully XML Schema-compliant.

REFERENCES[1]

[1] MPEG-7 Requirements Group, "MPEG-7 requirements", Doc. ISO/MPEG N4320, Sydney MPEG Meeting, July 2001.

[2] ISO/IEC 15938-2:2001, "Multimedia content description interface – Part 2 Description definition language", Version 1.

[3] XML Schema Part 0: Primer, W3C Candidate Recommendation, 24 October 2000. *http://www.w3.org/TR/xmlschema-0/*.

[4] XML Schema Part 1: Structures, W3C Candidate Recommendation, 24 October 2000. *http://www.w3.org/TR/xmlschema-1/*.

[5] XML Schema Part 2: Data types, W3C Candidate Recommendation, 24 October 2000. *http://www.w3.org/TR/xmlschema-2/*.

[6] MPEG7 Requirements Group, "Results of MPEG-7 Technology Proposal Evaluations and Recommendations", Doc. ISO/MPEG N2730, Seoul MPEG Meeting, March 1999.

[7] Extensible Markup Language (XML) Version 1.0 (Second Edition), World Wide Web Consortium Recommendation, 6 October 2000. *http://www.w3.org/TR/REC-xml*.

[8] J. Hunter, DSTC, "A Proposal for an MPEG-7 DDL", P547, MPEG-7 AHG, Test and Evaluation Meeting, Lancaster, February 1999.

[9] XML Schema Working Group. *http://www.w3.org/XML/Group/Schemas.html*.

[10] XML Linking Language (XLink) Version 1.0, W3C Recommendation, 27 June 2001. *http://www.w3.org/TR/xlink/*.

[11] XML Path Language (XPath) Version 1.0, W3C Recommendation, 16 November 1999. *http://www.w3.org/TR/xpath/*.

[12] ISO/IEC 15938-5:2001, "Multimedia content description interface – Part 5: Multimedia Description Schemes", Version 1.

[13] MPEG-7 DDL Response to XML Schema Last Call for Review, May 2000. *http://archive.dstc.edu.au/mpeg7-ddl/issues.html*.

[14] Namespaces in XML, W3C Recommendation, 14 January 1999. *http://www.w3.org/TR/REC-xml-names/*.

[15] ISO 8601 Draft Revision, ISO (International Organization for Standardization). Representations of dates and times, draft revision, 2000.

[16] RFC 2396 T. Berners-Lee *et al.*, RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax, 1998. *http://www.ietf.org/rfc/rfc2396.txt*.

[17] RFC 1766 H. Alvestrand, ed. RFC 1766: Tags for the Identification of Languages, 1995. *http://www.ietf.org/rfc/rfc1766.txt*.

[18] Resource Description Framework (RDF) Schema Specification 1.0, W3C Candidate Recommendation 27 March 2000. *http://www.w3.org/TR/rdf-schema/*.

[19] DAML + OIL (March 2001) *http://www.daml.org/2001/03/daml+oil-index.html*.

[1] Although MPEG documents are in principle nonpublic documents, many of them are made public and can be accessed at the MPEG Home Page, *http://mpeg.telecomitalialab.com/*. Nonpublic MPEG documents may be obtained through the MPEG Head of Delegation of the respective country.

5

Binary Format

Jörg Heuer[1], Cedric Thienot[2] and Michael Wollborn[3]

[1]Siemens AG, Munich, Germany, [2]Expway, Paris, France, [3]Robert Bosch GmbH, Hildesheim, Germany

5.1 OVERVIEW

In this chapter, the 'Binary Format for MPEG-7 Description Streams' (BiM) is described. As explained in Chapter 3, there are two possible ways to transmit MPEG-7 descriptions – either in textual form or in binary form. While both formats, the Textual Format for MPEG-7 (TeM) and the BiM, provide similar functionality with respect to *dynamic and incremental transmission* of description trees, the BiM provides an additional feature: compression of the verbose Extensible Markup Language (XML) representation used by TeM. In addition, the compressed binary format is designed in a way that allows fast parsing and filtering on binary level, without decompressing the complete description stream beforehand. This feature is, in particular, important for small, mobile, low-power devices with restricted CPU and memory capabilities.

As it is the case for the TeM, the BiM *Fragment Update Units,* building an *Access Unit*, also comprises three main parts: the *Fragment Update Command*, the *Fragment Update Context* and the *Fragment Update Payload*. The principle how the command and the context are used in order to update a description with the payload has already been presented in Chapter 3, since it is common for both TeM and BiM. This chapter describes how these elements are defined in the binary form and how they are decoded. Before these detailed descriptions, a short overview of the basic principles of BiM is given in the following text.

The encoding and decoding of the *Fragment Update Command* is quite simple. Since there are only four commands possible, each of them is represented by a fixed bit pattern. The representation of the *Fragment Update Context* is more complex, since in principle each node of a description tree can be selected as the *context node*. Therefore, as a first prerequisite, the decoder needs to know the definition of the descriptor or description scheme to which the MPEG-7 description corresponds. This definition is specified using the MPEG-7 *Definition Description Language* (DDL), which has been introduced in Chapter 4. The DDL is mainly based on the XML-schema language [1], so

the BiM is 'schema-dependent'. This means that a decoder needs to know the schema definition in order to decode a BiM stream and, for example, convert it into an XML document.

The construction of the *Fragment Update Context* can be interpreted as the specification of a path in the description tree, which identifies the *context node* that has to be updated. The addressing in the path relies upon schema knowledge, that is, the shared knowledge of encoder and decoder about the existence and position of potential elements within the description tree. The path specifies a node within a (virtual) description tree built from the possible – and not necessarily instantiated – elements as defined in the schema. In this description tree each node has a specific and fixed address which allows an unambiguous identification not depending on the current description as present at the decoder. The path itself is built by a concatenation of special codes, described in Section 5.2.2, which can address the children of a node passed in the path, its father, the type of a child node and (in case of possible multiple occurrences of nodes of the same declaration) the position of the child node. There are two *Context Modes* for specifying the *context node*: an absolute mode and a relative mode. Both modes differ by the origin of the path to the *context node*, as shown in Figure 5.1.

In the *Absolute Context Mode*, the origin of the *Context Path* is the so-called 'Selector node', which is a kind of virtual node at the very top of the description tree. The concept of the selector node is described in more detail later; at this stage, it is only important to know that it contains only one child element: the outermost element of the description. It can be interpreted like an absolute reference, and it does not change throughout the decoding of the description stream. In the *Relative Context Mode*, the origin of the *Context Path* is the *current context node* of the decoder. This node is defined by the *Context Path* of the previously decoded *Fragment Update Unit*. It can be interpreted as the status of the decoder, which can change after the decoding of each *Fragment Update Unit*. While the relative mode allows a very compact representation of the context, the absolute

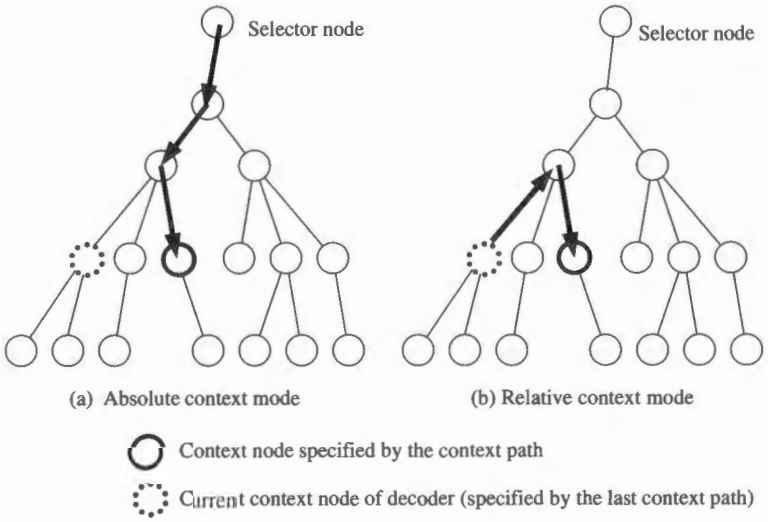


Figure 5.1 Interpretation of the Fragment Update Context as path to the context node

mode can be used to enable resynchronization or filtering of the binary description stream. Both modes can be arbitrarily combined within one description stream.

In each *Fragment Update Unit*, the *Fragment Update Payload* is transmitted as the final element (except for the case of a 'DeleteContent' or 'Reset' command). It should be noted that here payload does not necessarily mean the 'data', that is, the values of the description elements, but the payload can also contain structure elements. Therefore, in general, payload can be seen as a complete subtree of the description. This subtree is added to the *Current Description Tree*, or it replaces a subtree of the *Current Description Tree*. The position of the subtree to be added or replaced is specified by the *Fragment Update Context* described above.

The update payload coding is based on a similar idea than the coding of the update *Context Path*. In a very simplified and short view, it works as follows: starting at the top node of the subtree to be coded, a path in depth-first order[1] is followed. This path is now encoded, taking into account the knowledge of the schema that corresponds to the description. For example, the presence of mandatory nodes is not encoded, since it is clear that the node must be present in the subtree. For optional nodes and for nodes with multiple occurrences, special codes are assigned in order to signal the presence and (in case of multiple occurrences) the number of these nodes. These codes are generated by the encoder and the decoder, using the schema definition of the description. On the basis of this coding principle, the generated codes are quite compact, in particular, compared with the verbose names of XML-tags, and a very good compression for the description structure is achieved.

Since the type of an element is known to the encoder and the decoder from the schema, no further type information needs to be transmitted in general. Only for polymorphism elements, which may have one of several types, the instance type information is included in the bit stream.

A leaf element is an element that contains no further children but only the description data. The BiM compresses the description data of such leaves. Since different types of data values can occur, type-dependent coding is applied. This means that if the data type of the node is, for example, an 8-bit unsigned integer, then 8 bits are used in order to represent the value. In case of decimal or string values, the Unicode Transformation Format (UTF)-8 [2] representation is used. Special data types such as lists, matrices and so on are encoded by the concatenation of the values according to some specific rules described in Section 5.3. Because of the type-specific encoding, the resulting binary representation of the data values is more compact than the XML representation, which uses ASCII representation for all data values regardless of their type.

5.2 FRAGMENT UPDATE COMMAND AND CONTEXT

On the basis of the functionality of the BiM codec described above, different operating modes can be supported by a BiM encoder: successive coding in depth-first order,

[1] Depth-first order means that, starting from the top node of a tree or subtree, always the left most, not yet visited child of a node is visited until either a leaf node or a node whose children have all been visited is reached. Then, the path leads back to the father of that node, and from there the first rule is executed again. The path is finished when all nodes of the tree or subtree have been visited.

incremental coding in an arbitrary order and partial coding in which some elements of the tree or subtrees are transmitted separately. In any case, the order of encoding and transmitting of the elements can differ from the order in which the elements are specified for textual representations in the schema specification. To enable this flexible encoding, the *Fragment Update Context* of the *Binary Fragment Update Unit* is used. It identifies nodes or subtrees in the *Current Description Tree* on which the *Fragment Update Command* operates. This enables the decoder, for instance, to add a transmitted fragment at the correct context in the *Current Description Tree*. A detailed specification of *Fragment Update Commands* is given in Section 5.2.1.

To identify the context within the description tree, an absolute *Context Path* and a specification relative to the context of the previous *Fragment Update Unit* in the description is supported. These *Context Modes* and the encoding of the *Context Path* are described in more detail in Section 5.2.2.

5.2.1 Fragment Update Command

The *Fragment Update Command* is executed in the receiver by the *Description Composer* (see Section 3.3 in Chapter 3). The command 'AddContent' is used to transmit a description incrementally. Nodes or subtrees contained in these fragments are added to the current description in the decoder at the context identified by the *Fragment Update Context* (see Figure 5.2). The replacement of an existing node or subtree is supported by the 'ReplaceContent' command, for instance, to dynamically update a description of the score in a soccer game. The commands 'AddContent' and 'ReplaceContent' are followed by the BiM *Fragment Update Context* and the *Fragment Update Payload* that contains the encoded operand, that is, an attribute, element or subtree (see Section 5.3). If a part of such a description has become obsolete, the operand including its subtree can be deleted by the 'DeleteContent' command. For instance, the description of a player in a soccer game can be deleted after he received a match penalty and had to leave the playground. Contrary to the other commands, after the 'DeleteContent' command only a *Fragment Update Context* shall be contained in the *Fragment Update Unit*. To return to the initial state of the transmission, the 'Reset' command can be signalled in a *Fragment Update Unit* that consists only of this command. In this case, the decoder resets the description in the *Description Composer* to the state specified in the *Initial Access Unit* (see Chapter 3).

5.2.2 Fragment Update Context

The *Fragment Update Context* consists of three parts: the *Schema ID*, the *Context Mode* and the *Context Path*.

Schema ID

The *Schema ID* selects one schema specified in the *Decoder Initialization* (see Chapter 3). On the basis of this schema, the *Context Path* in this *Fragment Update Context* is encoded.

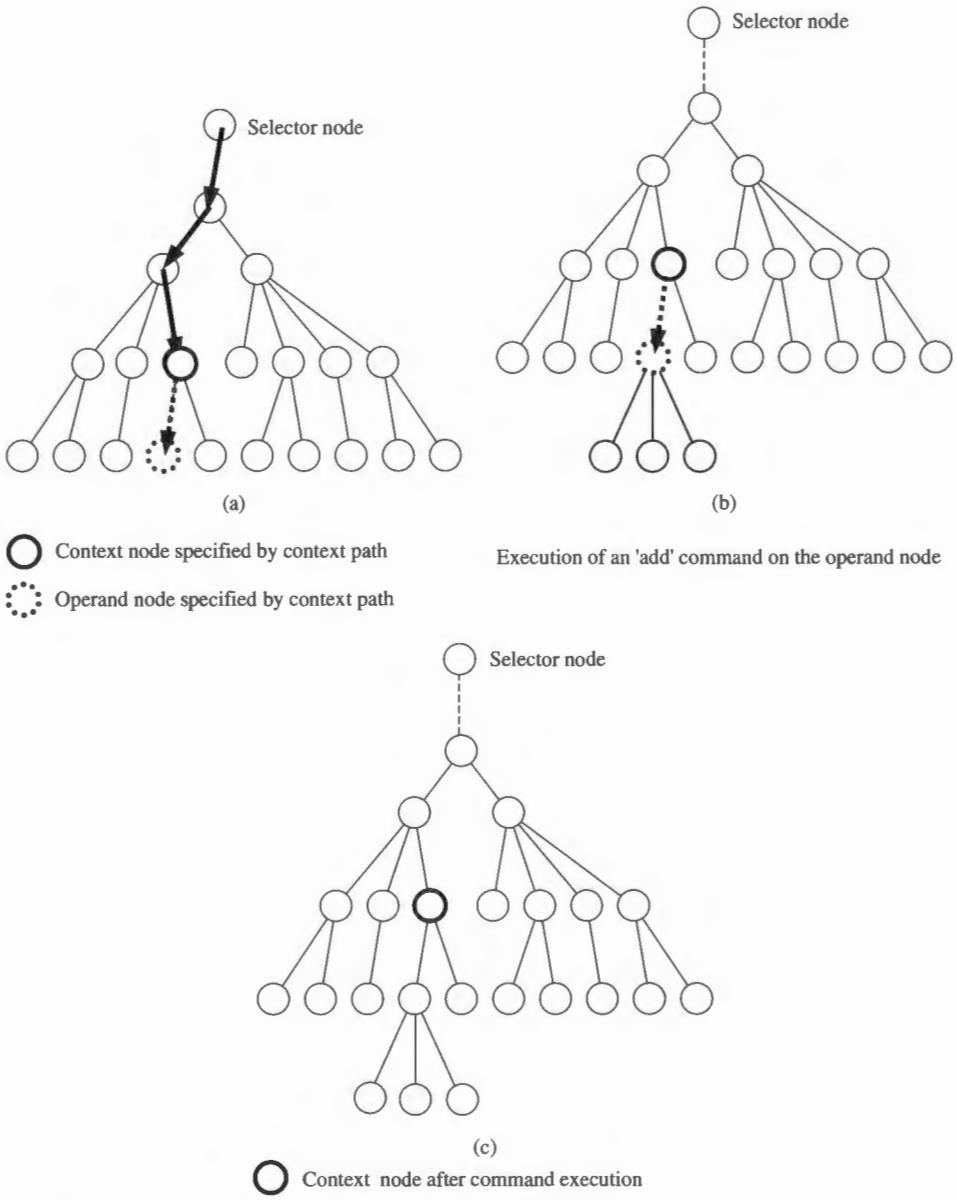


Figure 5.2 Processing of an 'AddContent' command in the *Description Composer*

Context Mode

Nodes in the description tree can be identified with an *Absolute Context Path* starting from the Selector Node of the description tree or with a *Relative Context Path* starting from the *current context node*. The *current context node* is specified by the previous *Fragment Update Unit* if a part of the description has already been transmitted (see Figure 5.1). To transmit several occurrences of a node in one *Fragment Update Unit*, both the described

Context Modes can be combined with the *Multiple Payload Mode* (see the paragraph on *Context Path* for details).

While the encoding with *Relative Context Paths* result in a very compact representation, the encoding with *Absolute Context Paths* do not depend on the decoding of the previously transmitted fragments. The *Absolute Context Path* enables the initialization of the BiM decoder to tune into an ongoing transmission, to filter *Fragment Update Units* with respect to the contained node or subtree or to resynchronize if the BiM decoder received an unknown (e.g. proprietary) extension of a description. To enable the decoder to skip unknown extensions, every *Fragment Update Unit* contains the *Skip Length* field at the beginning, which specifies the length of the *Fragment Update Unit* in bytes.

Using the *Multiple Payload Mode* can result in a very compact representation of elements or attributes of the same type that occur several times in the description. In this mode, several *Fragment Update Payloads* can be encoded in one *Fragment Update Unit*. For instance, this is a common case for certain visual descriptors such as the *Scalable Color Descriptor* (SCD), which can be instantiated several times to describe the color features in several frames of a video shot.

Context Path

The *Context Path* consists of two parts (see Figure 5.2):

- *The specification of the context node*: The *context node* is the context in which the *Fragment Update Command* operates, and the *current context node* for a subsequent *Context Path* if a *Relative Context Path* is specified in the next *Fragment Update Unit*.
- *The specification of the operand node*: The *operand node* is a child node of the *context node* and the node on which the *Fragment Update Command* is executed.

The nodes in the description tree of an MPEG-7 description are instantiated elements and attributes. Possible child nodes respectively content of these nodes are defined by type definitions in the schema. In MPEG-7, the schema definition is specified in ISO/IEC 15938 Part 5 (Multimedia Description Schemes (MDS), see Section III Chapter 6) [3], Part 3 (Video, see Section IV) [4], and Part 4 (Audio, see Section V) [5], which is known at the encoder and decoder end (see Chapter 3, Section 3.3). In contrast with XML Schema simple-type definitions, complex-type definitions can declare child elements and attributes. All nodes traversed in the path to the *context node* have to be an instantiated element of complex type. The *operand node* can be either an instantiated element of complex or simple type, an instantiated attribute or simple content[2]. This is due to the definition of an *operand node* to be a child node of the *context node*.

The *context node* in a description is identified by building a path through the description tree. The path consists of several context steps. A context step identifies a child node of complex type of the current node. The current node is identified by the previous context step. Depending on the *Context Mode*, the *context node* of the previous

[2] In the following, "node, which is an instantiated element of complex type" is abbreviated by "node of complex type" and "node, which is an instantiated element of simple type, an instantiated attribute or simple content of a complex type" is abbreviated by "node of simple type".

Fragment Update Unit or the selector node is set as current node at the beginning of the path. On the basis of this principle, all potential child nodes of complex type and the parent node of the current node have to be addressable by codes in a context step. In the following, these codes are called *Context Tree Branch Code* (Context TBC) and represent one context step in the path. The path is terminated by a termination code.

After the termination code is sent, the *operand node* is identified by a single operand step (Figure 5.2). In contrast with the *context node*, the *operand node* can also be of simple type. Accordingly, the operand step is signaled by a code of the Operand TBC table of the *context node*, which also includes codes to child nodes of simple type beside the codes to child nodes of complex type.

Tree Branch Codes

As stated above, a step in the path is specified by a TBC that identifies a child node of the current node. The current node is an instantiated element of complex type. All potential child nodes of the current node are the elements, attributes and simple content declared in the complex-type definition of the current node in the schema[3]. An example of a complex-type definition is given in Figure 5.3.

On the basis of the complex type definition in the schema, child elements and attributes of the current node in the description (see, for instance, Figure 5.4) can differ

```
<!--Definition of TextAnnotation datatype-->
<complexType name="TextAnnotationType">
      <choice minOccurs="1" maxOccurs="unbounded">
            <element name="FreeTextAnnotation" type="mpeg7:TextualType"/>
            <element name="StructuredAnnotation" type="mpeg7:StructuredAnnotationType"/>
            <element name="DependencyStructure" type="mpeg7:DependencyStructureType"/>
            <element name="KeywordAnnotation" type="mpeg7:KeywordAnnotationType"/>
      </choice>
      <attribute name="relevance" type="mpeg7:zeroToOneType" use="optional"/>
      <attribute name="confidence" type="mpeg7:zeroToOneType" use="optional"/>
      <attribute ref="xml:lang"/>
</complexType>
```

Figure 5.3 Example of a complex-type definition in a schema definition

```
...
<TextAnnotation relevance="1">
      <FreeTextAnnotation>
            ...
      </FreeTextAnnotation>
      <FreeTextAnnotation xsi:type="SubjectType">
            ...
      </FreeTextAnnotation>
      <KeywordAnnotation>
      ...
      </KeywordAnnotation>
</TextAnnotation>
```

Figure 5.4 Example of an instantiation of the TextAnnotation Element of TextAnnotationType in a description

[3] An exception is the selector node. In this case, the potential child nodes are all globally declared elements.

by their name, their type and their position with respect to siblings. Accordingly, a TBC is composed of

- the *Schema Branch Code* (SBC) and the *Substitution Code*, which identify the declaration of the child element;
- the *Type Code*, which identifies the type of the element if polymorphism is possible and;
- the *Position Code*, which identifies the position of the instantiated element within their siblings if the element or the group it is contained in can be instantiated more than once.

On the basis of this coding principle, a TBC table can be generated for every complex-type definition in the schema specification of the MPEG-7 standard. The TBC table used to encode or decode a step of the path in the description tree is defined by the complex type of the current node. After the step has been processed, the type and therefore also the TBC table of the 'new' current node is known. For instance, in Figure 5.5, a description tree is shown in which every node represents an element of the description. The type of a node is depicted in this figure in the node itself, while the name of the node is placed on the branch from the parent node. For simplicity it is assumed here that all TBCs only consist of SBCs for the instantiated complex types. That implies that all nodes can be instantiated only once and no polymorphism is applicable. Receiving an SBC, the decoder can map this information to the type of the child node on the basis of the schema definition (see Figure 5.5) and can successively decode the steps of the path.

In the following, the rules for the assignment of SBCs and *Substitution Codes*, *Type Codes* and *Position Codes* of TBCs based on complex-type definitions in the schema are explained.

SBCs and substitution codes

In Figure 5.3, an example of a complex-type definition is given. In the complex-type definition, possible child elements and attributes such as 'StructuredAnnotation' or 'relevance' are declared. To address all possible child nodes of a node of complex type, the

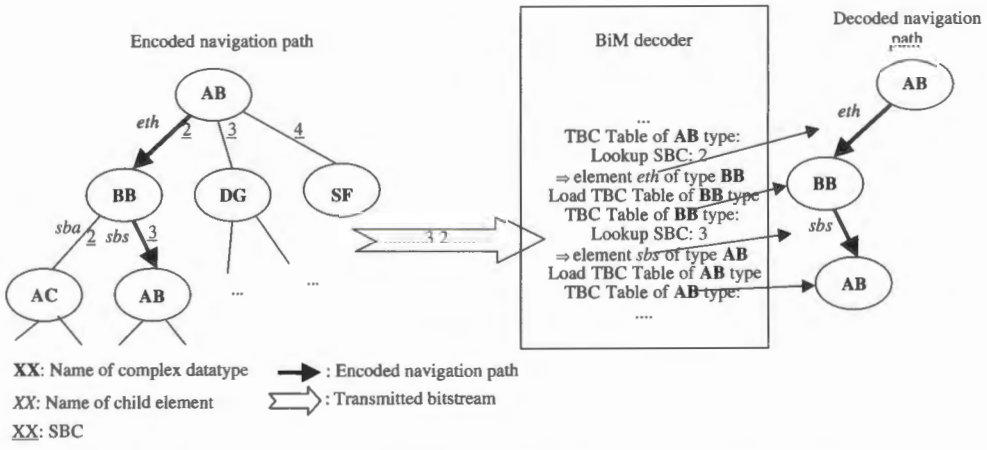


Figure 5.5 BiM Decoder processing an encoded navigation path of a description

Context TBC contains a *Context Schema Branch Code* (Context SBC), which identifies a child element of complex type[4] declared in the complex-type definition. Respectively, the bit length of the Context SBC in different Context TBC tables varies with the number of child element declarations that have to be addressed in the complex-type definition. For the complex-type definition in Figure 5.3, the Context SBCs are specified in Table 5.1. A detailed specification of the rules assigning SBCs is given in the MPEG-7 Systems standard [6].

Besides the SBC, a TBC can contain a *Substitution Code*. A substitution of a child element is possible if a global element is referenced in the complex-type definition, which is the head element of a substitution group [1]. The presence of a substitution is signaled in these TBCs by a flag and followed by a *Substitution Code* if the flag is set. The *Substitution Codes* are assigned to the elements in the substitution group in lexicographical order [6].

In addition to the Context SBCs that are assigned according to the complex-type definition, two generic SBCs are present in every Context TBC table: the *Reference to the Parent* and the *Path Termination Code*. To enable upward navigation, the 'all zero Context SBC' of every Context TBC table addresses the branch to the parent node of the current node in the description. To signal that the current node is the *context node*, the 'all one Context SBC' is defined as the termination code.

In contrast with Context TBC tables, Operand TBC tables contain Operand SBCs that refer also to child nodes of simple type but do not contain a termination code or a reference to the parent node. These child nodes can be elements or attributes of simple type or simple content as these are declared in the complex-type definition in the schema. For elements, the Operand TBCs are generated identically to Context TBCs identifying elements of complex type. In case of attributes, the Operand TBCs only consist of an Operand SBC because an attribute cannot be substituted or instantiated several times

Table 5.1 Context and operand TBC table for the example in Figure 5.3

Context and operand TBCs of TextAnnotationType					Tree Branch
Context SBC	Operand SBC	Substitution Code	Type Code	Position Code	
000	–	–	–	–	Reference to parent
–	0000	–	–	[Pos.Code]	User Data Extension Code
001	0001	–	–	[Pos.Code]	DependancyStructure
010	0010	–	[Type Code]	[Pos.Code]	FreeTextAnnotation
011	0011	–	–	[Pos.Code]	KeywordAnnotation
100	0100	–	–	[Pos.Code]	StructuredAnnotation
–	0101	–	–	–	Confidence
–	0110	–	–	–	Relevance
–	0111	–	–	–	xml:lang
101–110	1000–1111	–	–	–	
111	–	–	–	–	Path Termination Code

[4] Throughout the MPEG-7 schema, complex types are named starting with a capital letter. Accordingly, all declared elements in this example are of complex type, while the declared attributes are of simple type.

nor polymorphism can be applied. The 'all zero Operand SBC' is assigned to user data extensions as generic SBC in every Operand TBC table.

As an example, Table 5.1 lists the TBCs for child nodes of the TextAnnotationType. Since the Context and Operand TBCs differ only with respect to the Context and Operand SBCs defined, both are shown in a combined table in Table 5.1. According to the described functionality, all children of a node of the 'TextAnnotationType' are identified by an Operand SBC. The child elements of complex type are also addressed by a Context SBC.

Type Codes

In a textual description, instantiated elements can be of a type that is derived from the type in the element declaration of the schema. In the description, the 'applied' type is specified in an *xsi:type* attribute (see Section 4.3.4, Chapter 4 and the example in Figure 5.4). Such a 'type cast' is possible because of polymorphism feature in DDL and has to be signaled to the decoder to determine the correct TBC table of the subsequent 'current node'. In Figure 5.6, for a given type A, the inheritance tree is depicted. The inheritance tree is built on the basis of the type definitions in the schema. In the example of Figure 5.6, there are types such as 'AA' that are derived directly from type 'A'. Other types such as 'AAB' are derived in several steps from type 'A'. If the data type of a child element[5] is a base type for other data types, then a flag in the TBC specifies the presence of a type cast in the instantiation. If a type cast is present in the instantiation, the flag is followed by the *Type Identification Code* of the instantiated type. The *Type Identification Code* identifies a type within the inheritance tree of the base type of the encoded child element (see Figure 5.6). The *Type Identification Codes* are applied in depth-first order after ordering each sublevel of the inheritance tree lexicographically [6].

In the example of Table 5.1, the FreeTextAnnotation element requires the coding of *Type Codes* as the type of this element is assumed to be a base type from which another data type is derived (see Figure 5.7).

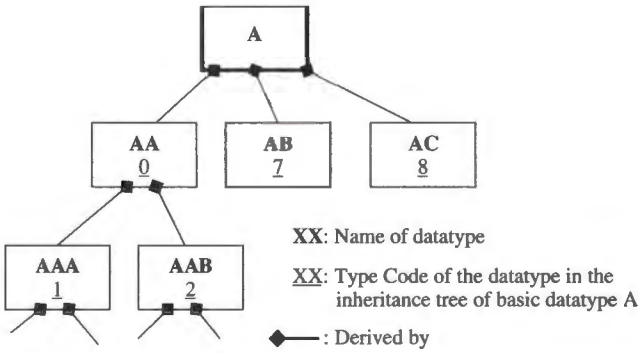


Figure 5.6 Type Codes for data types in the inheritance tree of basic type A

[5] XML Schema does not allow polymorphism in the case of attributes. Accordingly, this rule only applies to child elements.

```
<!--Definition of derived datatype -->
<complexType name="SubjectType">
      <simpleContent>
            <extension base="mpeg7:TextualType">
                  ...
            </extension>
      </simpleContent>
</complexType>
```

Figure 5.7 Example of a complex-type definition in a schema definition that is derived from a base type

Position Codes

Nodes and subtrees of the description tree can be transmitted in an arbitrary order if those are encoded in fragments. The *Context Path* specifies the context in which the fragment is added to the description tree. The SBC, the *Substitution Code* and the *Type Code* in the TBCs are sufficient to position the element or subtree within the description tree, if all child elements in the encoded path can only be instantiated once. A *Position Code* has to be present in every TBC if a multiple instantiation of a child element[6] is possible in the description tree based on the schema definition. For the encoding of the position number, two cases are distinguished: the complex-type definition permits (1) multiple occurrence of model groups, for example, choices of child elements (see the maxOccurs attribute in Figure 5.3) or (2) only multiple occurrence of a child element[7]. In case (1), the position number of a specific child element is encoded globally counting the position with respect to all child elements. In case (2), the position number is encoded only with respect to the child element with multiple occurrences that results in a shorter *Position Code*. In both cases, if the number can exceed a value of 16, the position number is encoded with variable length [6]. In the example of Table 5.1, all child elements require to signal global *Position Codes* with variable length because they are contained in a choice group that can be instantiated as often as needed (case (1)).

In case the *Context Mode* is set to the *Multiple Payload Mode* with relative or absolute *Context Path*, the *Position Codes* of the initial transmitted *Context Path* in this *Fragment Update Unit* can be incremented. Therefore, in one *Fragment Update Unit*, a command can be executed on several nodes in the description tree, which differ in their *Context Path* only by the *Position Codes*. For each of these nodes also, a *Fragment Update Payload* is contained in this *Fragment Update Unit*.

Data partitioning of the Context Path

A feature of the binary *Context Path* representation is to enable a fast filtering of binary *Fragment Update Units* on the basis of the context of the contained *Fragment Update Payload*. The context characterizes the information in the payload. On the basis of this information, the filter considered here can decide if desired information can be contained in a certain *Fragment Update Unit*. The fast filtering can be applied in the binary domain

[6] Because of the fact that attributes can only have one occurrence, this rule only applies to elements but not to attributes.

[7] In this case, only in the declaration of the child element the maxOccurs attribute is specified with a value greater than one.

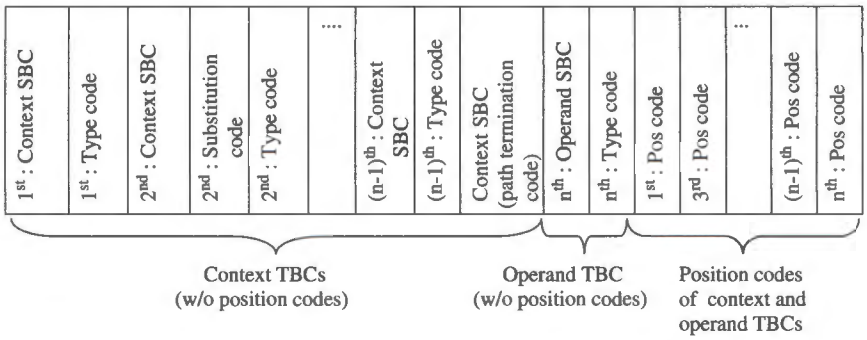


Figure 5.8 Structure of a Context Path in the MPEG-7 BiM bitstream (a number is assigned to each TBC)

on absolute *Context Paths* that identify the node or subtree in the transmitted *Fragment Update Payload*. That is, no decoding of the binary *Context Path* is needed for this filtering. Instead, bit patterns are generated by encoding *Context Paths* that specify the filtering rules. To enable the specification of these predefined bit patterns for a variety of filtering rules, the binary representation of the *Context Path* is partitioned into two parts (see Figure 5.8):

- the first part that contains the SBCs, *Substitution Codes* and *Type Codes* of the TBCs building the Context Path and
- the second part that consists of all *Position Codes* in the same order as the respective codes of the TBCs are signaled in the first part.

This data partitioning allows the generation of a bit pattern of fixed length for filtering of certain *Fragment Update Payloads* in the binary domain, even if the *Position Codes* are not known. In this case, the first part of the representation of such a 'binary filter rule' is of fixed length, whereas the second part of the *Context Paths* can vary in length (see the section on *Position Codes*).

For instance, assume that a filter rule is specified to filter the creation information of video clips. Therefore, all described video clips should be filtered and only a specific one should not be described in the transmitted description at a known position. The position codes of the video segment descriptions can vary in length. Therefore, it would not be possible to specify a bit pattern for filtering if the *Position Codes* are interleaved within the TBCs. But because of the data partitioning, a bit pattern specifying only the first part of the binary *Context Path* can be used to specify this filter rule in the binary domain.

5.3 BINARY PAYLOAD REPRESENTATION

An XML Document is a tree of nodes (mainly composed of elements, attributes and data). Its textual exchange format is well known and defined by the W3C [7]. This section presents a binary exchange format of an XML Document called *Binary Payload Representation* and is used to encode the *Fragment UpdatePayload*. This format generates a binary encoding of an XML document on the basis of its schema definition, which 'in

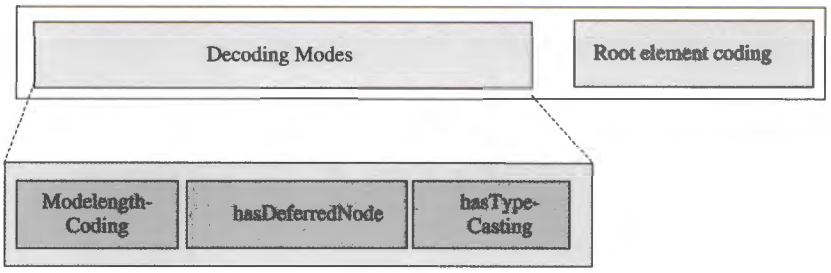


Figure 5.9 Principle structure of the binary payload representation

XML' means both validity constraints for the textual representation and binary encoding for the binary level. This binary format keeps the 'XML spirit' by providing the ability to support standard XML APIs.

5.3.1 General Overview

The binary format is composed of one global header, the *Decoding Modes*, which specifies some general parameters of the encoding, and a set of consecutive and nested coding patterns (see Figure 5.9). These patterns are nested in the same way the elements are nested in the original XML file (i.e. as a tree). As only one root element is allowed in an XML document, all the patterns are contained in one large root element-coding pattern.

Element coding

An element-coding pattern is used to code XML elements. Each element coding is composed of a 'length', a *Substitution Code*, which encodes the XML-Schema substitution information, a *Type Code*, which encodes the XML-Schema type information, an 'attributes' field and a 'content' field.

The general form of an element-coding pattern is the following (Figure 5.10):

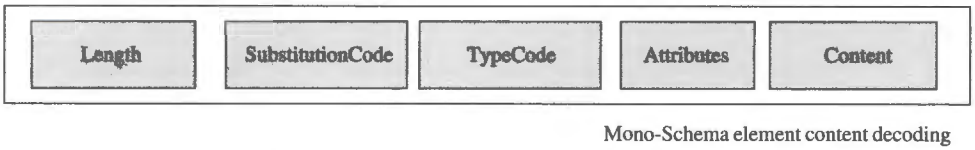


Figure 5.10 Principle structure of an element-coding pattern

Length

The length field specifies the coding length in bits of the element. This feature allows a decoder to skip the entire element, saving CPU. Its presence can be optional, mandatory or forbidden according to the *Decoding Modes* specified in the header.

Substitution Code

XML schema allows some elements to be substituted by other elements. Such substitution is constrained to elements belonging to the same substitution group [1]. For such elements,

a flag codes if the substitution occurs. If this flag is equal to *true*, a code of the substitute element is signaled. This code is built using all the possible substitute elements.

Type Code

The *Type Code* is used to code subtyping information, nil and partially instantiated elements:

- *Subtyping*: The specific attribute 'xsi:type' defined in XML schema allows to change the type of an element, directly within the description. The *Type Code* is built using the set of all the possible subtypes.
- *Nilelement*: The type code allows to specify that a type is nil (cf. [1]).
- *Deferred Nodes*: Deferred Nodes allow to cut description trees or subtrees to be encoded into pieces. In this case some elements of the subtree are deferred. This means that their value and attributes are not present in the current *Fragment Update Payload* but will be sent later. In order to encode this feature, a specific type 'uncoded' is artificially added at the first place of the set of all possible subtypes.

For instance, the *Type Code* is absent if the type has no subtype, if it is not 'nilable' and if Deferred Nodes are forbidden in the payload (according to the *Decoding Modes*).

Attributes

The attributes are coded as a set of consecutive patterns. These patterns are composed of two components:

- *The attribute presence flag*: This flag is present if the attribute is optional; it indicates if the attribute is coded or not;
- *The attribute value*: The value of the attribute is coded according to the data type of the attribute.

Before being encoded, the following rules apply:

- All the attributes defined as 'fixed' or 'prohibited' are suppressed;
- The attributes are lexicographically sorted according to their Expanded Name, that is, the concatenation of the attribute name space and name, separated by a ':' character.

Content

The content of an element is either a simple type or a complex type. The decoding processes are described in Sections 5.3.2, 5.3.3.

5.3.2 Complex-Type Coding

Finite-state automaton decoders

The decoding and encoding process of complex type is performed by a set of 'finite-state automaton decoders' built according to the schema definition of the complex type. These automata are constructed according to the structure of its content model (a sequence, a

choice of elements, a sequence of choices, ...) and the cardinality of the content model components ('minOccurs' and 'maxOccurs' attributes).

For instance, the following type *t*:

```
<complexType name="t">
      <element name="a" type="ta" minOccurs="0" maxOccurs="1"/>
      <choice minOccurs="1" maxOccurs="1">
            <element     name="b"    type="Tb"    minOccurs="1"
                       maxOccurs="unbounded"/>
            <element name="c" type="Tc" minOccurs="1" maxOccurs="1"/>
      </choice>
</complexType>
```

is decoded by the following finite-state automaton decoder (Figure 5.11):

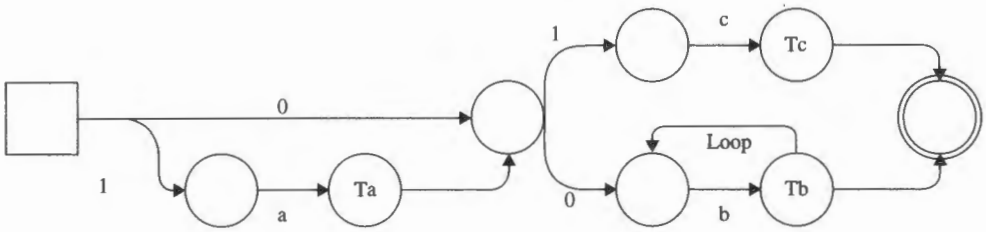


Figure 5.11 Example of an automaton decoder

Decoding a complex type using its finite-state automaton decoder

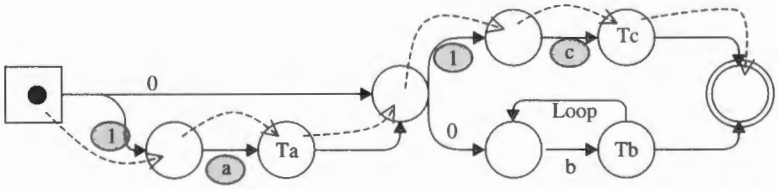
The decoding of a complex type is done by the propagation of a token through its associated automaton. When the token can follow different directions (i.e. when it can reach different states), it reads the bitstream to find the right way to go. Codes are assigned to each transition.

Lets consider the following bitstream:

```
Bit stream
```

1 [a] 1 [c]

```
Path followed by a token during the decoding of a complex type
```




```
resulting decoded description

        <Y>
          <a>...</a>
          <c>...</c>
                                    </Y>
```

The number of occurrences in the description is coded according to the minimal and maximal number of occurrences defined in its schema definition. In the example, b is the only element whose number of occurrences is encoded.

Therefore, the two following examples will be encoded as:

XML	Content code of *Y*						
⟨Y⟩ ⟨a⟩ . . . ⟨/a⟩ ⟨b⟩ . . . ⟨/b⟩ ⟨b⟩ . . . ⟨/b⟩ ⟨/Y⟩	1	[a]	0	**2**	[b]	[b]	
⟨Y⟩ ⟨b⟩ . . . ⟨/b⟩ ⟨b⟩ . . . ⟨/b⟩ ⟨b⟩ . . . ⟨/b⟩ ⟨/Y⟩	0		0	**3**	[b]	[b]	[b]

Generation of finite-state automaton decoders

The finite-state automata process is composed of four phases (Figure 5.12):

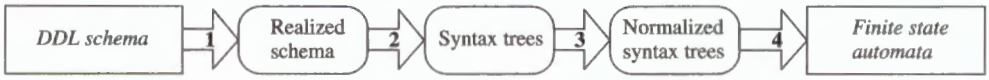


Figure 5.12 Phases of the finite-state automata process

Phase 1 – Schema realization

This phase flattens type inheritances. It realizes group references, element references, possible substitutions, possible subtyping and imports.

Phase 2 – Syntax trees generation

This phase produces a syntax tree for every complex content. These trees are transformed in order to improve compression ratio. For instance, the previous type *t* produces the syntax tree as shown in Figure 5.13.

Phase 3 – Syntax trees normalization

This phase normalizes the syntax trees in order to produce a signature for every tree node. These signatures are used in phase 4 to unambiguously generate codes. For instance, in the previous type *t*, the signature of the choice group is the string 'choice b c'.

Phase 4 – Finite-state automaton decoders generation

This final phase produces the finite-state automaton decoders used during the decoding (cf. Figure 5.10 – an example of an automaton decoder).

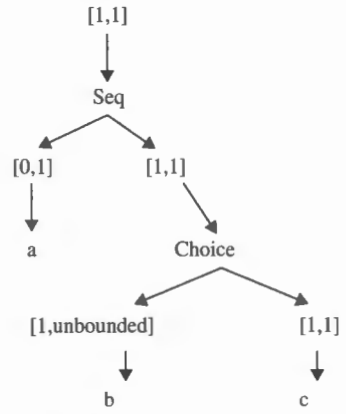


Figure 5.13 Example of a syntax tree

5.3.3 Simple-Type Coding

In MPEG-7, most of the data types are coded using well-known encoding standards (for instance, float is coded as an IEEE 754 floating-point 'single precision'). Nevertheless, some particular features of XML schema have been used to improve data type encoding, that is:

- *List*: Each item of a list is coded using its data type encoder. The number of items is coded at the beginning;
- *Integer*: Constrained integer decoder uses the 'minInclusive', 'maxInclusive', 'minExclusive' and 'maxExclusive' facets to deduce the minimum coding length of an integer;
- *Enumeration*: Every enumerated data type is encoded according to a sorted dictionary of its possible values.

It is also possible to specify a codec and to associate it with a particular type of schema in the *Decoder Initialization* (see Chapter 3).

5.3.4 Extensions and Forward/Backward Compatibility

Further versions of MPEG-7 description schemes are possible (versioning); moreover, it is possible to define a private schema based on MPEG-7 description schemes and descriptors. These evolutions will be made in the DDL framework by designing a new schema from an old one. The binary format supports the backward and the forward compatibility. This section describes how this functionality is handled by the BiM format. The main idea is to cut the coding of an element into schema-dependent pieces. Therefore, a decoder will be able to skip encoded parts using an unknown schema.

In a description, it is possible to change the schema by using either the polymorphism mechanism or the substitution mechanism. In this section, only the polymorphism case is presented. Substitution handling can be found in [6].

The Binary Payload Representation allows a partial interpretation (or decoding) of a received description, although needed schemas are not available. For that purpose, the encoder uses type hierarchy information to generate a compatible binary format.

Lets consider a simple example: a type T2 is defined in a schema S2 as an extension of a type T1, defined in a schema S1. According to XML-schema rules, the effective content model of T1 is composed of two parts, the first part coming from S1 and the second part coming from S2.

Defined in S1:

```
<complexType name="T1">
      <element name="x" type="integer"/>
</complexType>
```

Defined in S2:

```
<complexType name="T2">
      <extension base="S1:T1">
            <element name="y" type="integer"/>
      </restriction>
</complexType>
```

Actually, the 'effective' content model of T2 is:

```
<complexType name="T2">
      <element name="x" type="integer"/> <!-- T1 part -->
      <element name="y" type="integer"/> <!-- T2 part -->
</complexType>
```

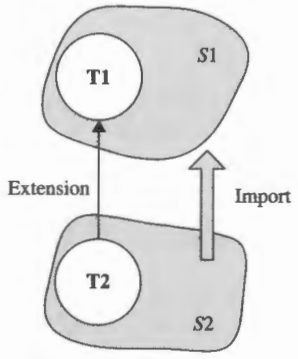


Figure 5.14 Example of extended schema

Let us consider the following description:

```
<anElement>
     <S1:x>10</S1:x>
     <S2:y>20</S2:y>
</anElement>
```

It could be encoded in two different ways:

- The first ensures the compatibility with an *S*1-decoder (an *S*-decoder can decode only descriptions valid according to the schema *S*).

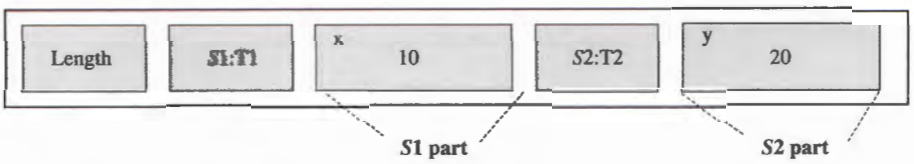


Figure 5.15 Example of encoding pattern compatible with *S*1 and *S*2

The binary encoding uses the *Schema ID* to separate these two schema-dependant parts (noted "S1:T1", Figure 5.15). Therefore, an *S*1-decoder will be able to decode the *S*1 part and skip the *S*2 part (using the length), while an *S*2 decoder will be able to decode both the parts.

- The second, more compact, does not preserve the compatibility with an *S*1-decoder: an encoder could choose to increase coding ratios but decrease interoperability between versions; it can therefore encode the element in the following way (Figure 5.16):

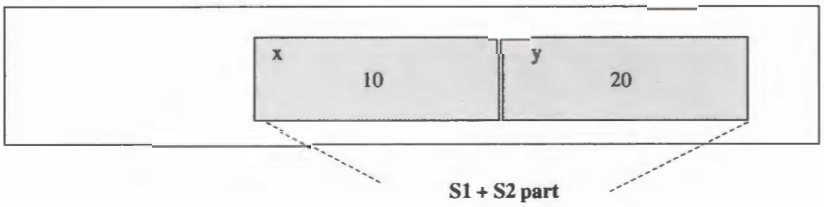


Figure 5.16 Example of encoding pattern compatible with *S*2

This multiple schema coding is not used for each element of the description. Actually, in order to improve the coding ratio, a schema mode is coded. This schema mode allows, for example, to freeze a schema in a subtree, that is, the overall subtree will be coded using only one schema.

5.4 CONCLUSION

The MPEG-7 binary description stream format, nicknamed BiM, provides a flexible and efficient tool for the compression and transmission of MPEG-7 descriptions. On the one hand it preserves the MPEG-7 Systems features of incremental transmission and dynamic update of descriptions, while on the other hand it provides a very compact representation compared with XML. Moreover, the format is designed in a way that allows fast search and filtering on binary level, that is, without the need for decompressing the incoming description stream. Finally, the BiM can not only compress and transmit MPEG-7 descriptions but it can also be applied to XML files in general, as long as they are based on DDL or XML schema and the respective DDL or XML-schema definition is available.

ACKNOWLEDGMENT

The MPEG-7 BiM presented in this chapter is the result of contributions and significant collaborative efforts of many MPEG-7 participants. The authors are grateful to all who contributed to the standardization efforts and to the editors of the MPEG-7 Systems specification. Special thanks go to Andreas Hutter, Ulrich Niedermeier, Claude Seyrat and Grégoire Pau for their help in preparing this chapter.

REFERENCES

[1] XML Schema Part 1: Structures. Available at: *http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/*.
[2] ISO/IEC 10646-1:1993, Amendment 2, Annex R, International Standard – Information technology – Universal Multiple-Octet Coded Character Set – Part 1: Architecture and Basic Multilingual Plane.
[3] ISO/IEC 15938-5:2001, "Multimedia Content Description Interface – Part 5 Multimedia Description Schemes", Version 1.
[4] ISO/IEC 15938-3:2001, "Multimedia Content Description Interface – Part 3 Visual", Version 1.
[5] ISO/IEC 15938-4:2001, "Multimedia Content Description Interface – Part 4 Audio", Version 1.
[6] ISO/IEC 15938-1:2001, "Multimedia Content Description Interface – Part 1 Systems", Version 1.
[7] XML. Available at: *http://www.w3.org/TR/2000/REC-xml-20001006*.

Section III
DESCRIPTION SCHEMES

6

Overview of Multimedia Description Schemes and Schema Tools

Philippe Salembier[1] and John R. Smith[2]

[1]*Universitat Politècnica de Catalunya, Barcelona, Spain,* [2]*IBM T. J. Watson Research Center, Hawthorne, New York, USA*

6.1 INTRODUCTION

The goal of the MPEG-7 standard [1] is to allow interoperable searching, indexing, filtering and access of multimedia content by enabling interoperability among devices that deal with multimedia content description. The standard provides four types of normative elements: Descriptors, Description Schemes, a Description Definition Language (DDL) and Coding Schemes. The MPEG-7 descriptors are designed to describe individual features of multimedia content. The description schemes provide complex descriptions by integrating together multiple descriptors and description schemes. The MPEG-7 DDL provides a language for defining the description schemes and descriptors. The DDL also allows the extension and modification of the MPEG-7 standardized description schemes and descriptors. Finally, the MPEG-7 Coding Schemes are designed for compressing the MPEG-7 textual XML descriptions in order to satisfy application requirements for compression efficiency, error resilience, random access, streaming and so forth. We briefly describe in the following text the major relations between descriptors, description schemes and DDL.

6.1.1 Descriptors

The MPEG-7 descriptors are designed for describing the following types of information: low-level audiovisual features such as color, texture, motion, audio energy and so forth; high-level features of semantic objects, events and abstract concepts; content management processes; information about the storage media and so forth. It is expected that most descriptors corresponding to low-level features will be extracted automatically, whereas human intervention will be required for producing the high-level descriptors.

6.1.2 Description Schemes

The MPEG-7 description schemes expand on the MPEG-7 descriptors by combining individual descriptors and other description schemes within more complex structures and by defining the relationships between the constituent descriptors and description schemes. In MPEG-7, the description schemes are categorized as pertaining specifically to the audio or visual domain, or pertaining generically to the description of multimedia. For example, typically, the generic description schemes correspond to immutable metadata related to the creation, production, usage and management of multimedia as well as to describing the content directly at a number of levels including signal structure, features, models and semantics. Typically, the Multimedia Description Schemes (MDSs) refer to all kinds of media consisting of audio, visual and textual data, whereas the domain-specific descriptors, such as those for color, texture, shape, melody and so forth, refer specifically to the audio or visual domain. As in the case of descriptors, the instantiation of the description schemes can, in some cases, rely on automatic tools but in many cases will require human involvement or authoring tools.

6.1.3 DDL

The descriptors and description schemes are defined using the MPEG-7 DDL [4], which is an extension of the XML Schema language [5]. An MPEG-7 description is produced for a particular piece of multimedia content by instantiating the MPEG-7 descriptors and description schemes. The resulting MPEG-7 description may take a textual form (XML), which is suitable for editing, searching and filtering, or a binary form (BiM) [3], which is suitable for storage, transmission and delivery.

The objective of this chapter is to provide a brief introduction to the (MDSs) being developed as part of the MPEG-7 standard. Most of these description schemes are discussed more thoroughly in Chapters 7–11 [6–10]. Furthermore, detailed information can be found in the Experimentation Model (XM) [11] and MPEG-7 Part 5 [12] documents being developed by the MPEG MDS Group. The structure of the chapter is as follows: Section 6.2 briefly reviews the organization of the MPEG-7 description schemes and highlights the most relevant aspects of the different classes of description schemes. Then, Section 6.3 describes in more detail the specific design and functionalities of the *Mpeg7* root and the top-level elements. Finally, Section 6.4 summarizes the MPEG-7 MDSs.

6.2 ORGANIZATION OF MDS TOOLS

In this section, we organize the presentation of the MPEG-7 descriptors and description schemes on the basis of their functionality. Figure 6.1 provides an overview of the organization of MPEG-7 MDSs into the following areas: Basic Elements, Content Description, Content Management, Content Organization, Navigation and Access and User Interaction.

6.2.1 Basic Elements

The MPEG-7 MDS specification defines a number of basic elements that are used repeatedly as fundamental constructs throughout the definition of the MPEG-7 description

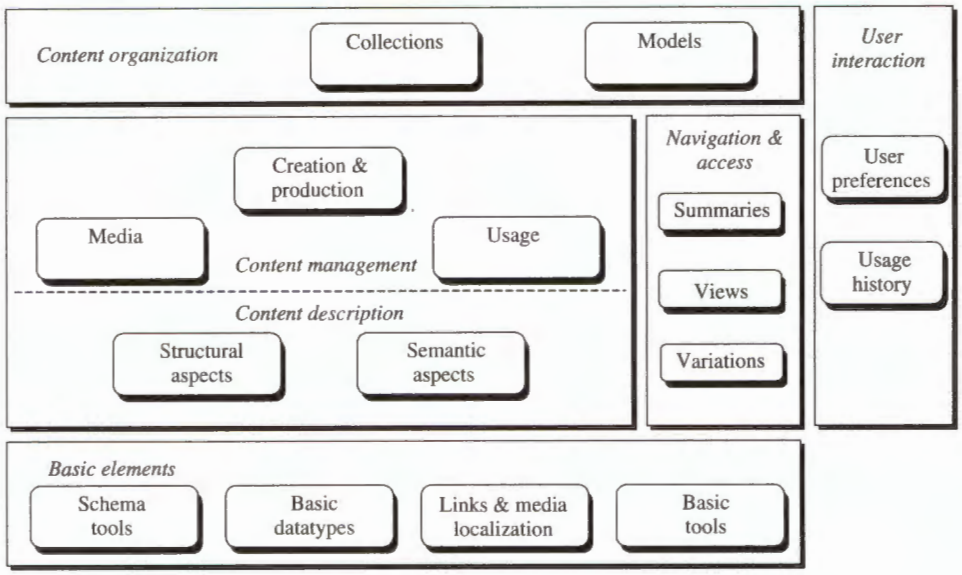


Figure 6.1 Overview of the MPEG-7 MDS

schemes. Basic elements include Schema tools, basic data types, Linking, identification and localization tools, as well as Basic description tools.

The MPEG-7 MDS specification defines a number of Schema Tools that facilitate the creation of valid MPEG-7 descriptions and packaging. The specification defines a base type hierarchy that organizes the descriptors and description schemes. The type hierarchy defines the base set of tools, including description schemes, descriptors and Header, from which all specific MPEG-7 tools are derived. The abstract types for descriptors and description schemes extend from the Mpeg7BaseType. The base type for description schemes (DSType) includes attributes that indicate time and media time information is specified in the description. The audio and visual descriptors and description schemes extend from the abstract DType (descriptors) and DSType (description schemes), respectively. The specification also defines the *Mpeg7* root and the top-level elements, which are used to form MPEG-7 valid descriptions. The root element and the top-level elements are described in more detail in Section 6.3 of this chapter.

The Schema tools specification also defines a Package tool that is used to describe an organization or packaging of specific descriptors and description schemes for an application. A Package can be used to organize and label the tools for ease of use and navigation. Packages provide a mechanism for conveying the structure and types of description information to aid users to overcome unfamiliarity. For example, a Package description can be communicated to a browsing tool to indicate structure or elements of the multimedia content descriptions. Packages also provide a mechanism for signaling between a database and query application about what description elements are available for querying.

Finally, the Schema tools specification defines a Description Metadata description scheme that is used to describe metadata about the description itself. The Description

Metadata description scheme can be embedded into a description tool to describe metadata concerning that description. Such metadata includes information about identifying the description (privately or publicly), the creation of the description, the version of the description and the rights associated with the description.

The MPEG-7 MDS specification also defines a number of basic elements that are used repeatedly as fundamental constructs throughout the definition of the MPEG-7 description schemes. Many of the basic elements provide specific data types and mathematical structures, such as vectors, and matrices, which are important for multimedia content description. Also included as basic elements are constructs for linking media files and localizing segments, regions and so forth. Many of the basic elements address specific needs of multimedia content description, such as the description of time, places, persons, individuals, groups, organizations and several forms of textual annotation. More information about these description schemes can be found in Chapter 7 [6].

6.2.2 Content Management

MPEG-7 provides also description schemes for content management. Together, these elements describe different aspects of (1) creation and production, (2) media coding, storage and file formats and (3) content usage. The functionality of each of these classes of description schemes is given as follows[1]:

- *Creation Information* describes the creation and production of the multimedia content. The Creation information provides a Title (which may itself be textual or another piece of multimedia content), Textual Annotation and information such as creators, creation locations and dates. It describes also how the multimedia material is classified into categories such as genre, subject, purpose, language and so forth. Furthermore, it provides review and guidance information such as age classification, subjective review and parental guidance. Finally, the Related Material information describes whether there exists other multimedia material that is related to the content being described.
- *Usage Information* describes information related to the usage rights, usage record and financial information. The rights information is not explicitly included in the MPEG-7 description. Instead, links are provided to the rights holders and to other information related to rights management and protection. The underlying strategy is to enable MPEG-7 descriptions to provide access to current rights owner information without dealing with information and negotiation directly. The Usage Record description schemes provide information related to the use of the content such as broadcasting, on demand delivery, CD sales and so forth. Finally, the Financial description scheme provides information related to the cost of production and the income resulting from content use. The Usage Information is typically dynamic in that it is subject to change during the lifetime of the multimedia content.
- *Media Description* describes the storage media in particular the compression, coding and storage format of the multimedia content. It identifies the master media that is the original source from which different instances of the multimedia content are produced.

[1] Many of the components of the DSs are optional. The instantiation of the optional components is often decided in view of the specific multimedia application.

The instances of the multimedia content are referred to as Media Profiles, which are versions of the master obtained perhaps by using different encoding, or storage and delivery formats. Each Media Profile is described individually in terms of the encoding parameters, storage media information and location.

6.2.3 Content Description

MPEG-7 provides also description schemes for content description. These elements describe the Structure (regions, video frames and audio segments) and Semantics (objects, events and abstract notions). The functionality of each of these classes of description schemes is given as follows:

- *Structural aspects* description schemes describe the multimedia content from the viewpoint of its structure. The description is built around the notion of Segment description scheme that represents a spatial, temporal or spatiotemporal portion of the multimedia content. The Segment description scheme can be organized into a hierarchical structure to produce a Table of Content for accessing or an Index for searching the multimedia content. The Segments can be further described on the basis of perceptual features using MPEG-7 Descriptors for color, texture, shape, motion, audio features and so forth, as well as semantic information using Textual Annotations.
- *Conceptual aspects* description schemes describe the multimedia content from the viewpoint of real-world semantics and conceptual notions. The Semantic description schemes involve entities such as objects, events, abstract concepts and relationships. The Segment description schemes and Semantic description schemes are related by a set of links that allows the multimedia content to be described on the basis of both content structure and semantics together. The links relate different Semantic concepts to the instances within the multimedia content described by the Segments.

Many of the individual description schemes for content description and content management are presented in more detail in Chapter 8 [7]. Note that, most of the MPEG-7 description schemes are linked together and in practice, the description schemes are included within each other in MPEG-7 descriptions. For example, Usage information, Creation and Production and Media information can be attached to individual Segments identified in the MPEG-7 description of multimedia content structure. Depending on the application, some aspects of the multimedia content description can be emphasized, while others can be minimized or ignored.

6.2.4 Navigation and Access

MPEG-7 provides also description schemes for facilitating browsing and retrieval by defining summaries, views and variations of the multimedia content.

- *Summaries* provide compact highlights of the multimedia content to enable discovery, browsing, navigation, visualization and sonification of multimedia content. The Summary description schemes involve two types of navigation modes: hierarchical and sequential. In the hierarchical mode, the information is organized into successive levels, each describing the multimedia content at a different level of detail. In general, the

levels closer to the root of the hierarchy provide more coarse summaries and levels further from the root provide more detailed summaries. The sequential summary provides a sequence of images or video frames, possibly synchronized with audio, which may compose a slide show or multimedia skim.

- The *View DS* is based on *Partitions* and *Decompositions*, which describe different decompositions of the multimedia signals in space, time and frequency. The partitions and decompositions can be used to represent different views of the multimedia content, which is important for multiresolution access and progressive retrieval.
- *Variations* provide information about different variations of multimedia programs, such as summaries and abstracts; scaled, compressed and low-resolution versions and versions with different languages and modalities – audio, video, image, text and so forth. One of the targeted functionalities of the Variation description scheme is to allow the selection of the most suitable variation of a multimedia program, which can replace the original, if necessary, to adapt to the different capabilities of terminal devices, network conditions or user preferences.

The Navigation and Access description schemes are described in more detail in [8].

6.2.5 Content Organization

MPEG-7 provides description schemes for organizing and modeling collections of multimedia content. The Collection description scheme organizes collections of multimedia content, segments, events, objects or even content descriptions. This allows each collection to be described as a whole on the basis of the common properties. In particular, using the Model description scheme, different models and statistics may be specified for characterizing the attribute values of the collections. The Content Organization description schemes are described in more detail in Chapter 10 [9].

6.2.6 User Interaction

Finally, the last set of MPEG-7 description schemes deals with User Interaction. The User Interaction describes user preferences and usage history pertaining to the consumption of the multimedia material. This allows, for example, matching between user preferences and MPEG-7 content descriptions in order to facilitate personalization of multimedia content access, presentation and consumption. The main features of the User Interaction description schemes are described in Chapter 11 [10].

6.3 SCHEMA TOOLS: *MPEG7* ROOT AND TOP-LEVEL ELEMENTS

As illustrated in Figure 6.1, Schema tools are considered as basic elements. Compared with other basic elements, Schema tools have a different functionality because they do not target the description of the content but are used to create valid descriptions and to manage them. As a result, we present here the most important Schema tools and discuss the remaining basic elements allowing content description in Chapter 7 [6].

The *Mpeg7* root element and the top-level elements are specific elements used to form MPEG-7 valid descriptions. The root element is the main element enclosing an entire description. In this sense, the root element serves as a wrapper for an MPEG-7 description. It also specifies the model of valid descriptions. Two types of valid descriptions are distinguished: complete descriptions and partial descriptions that are instances carrying partial or incremental information for an application, which are called Description Units. The Mpeg7 root element is illustrated in Figure 6.2 where notation is based on the Unified Modeling Language (UML) [2]. Each rectangular box corresponds to a description scheme or descriptor; paths between boxes denote composition relationships and strings such as '1,*' indicate the lower and upper bounds on the multiplicity of the relationship. As reported in Figure 6.2, a Description Unit can include any MPEG-7 element: Descriptor or description scheme. They should be used for situations in which only an elementary piece of information is required. Complete descriptions usually contain a 'semantically-complete' description for a given application. In the case of complete description, the top-level elements appear immediately following the root element and selectively include tools of the MPEG-7 schema that are appropriate for particular description tasks. The top-level types serve as wrappers of tools for particular description tasks. There are three major description tasks: Content Entity Description, Content Abstraction Description and Content Management (light gray elements in Figure 6.2).

- *Description of ContentEntity*: These elements provide a model for describing different types of multimedia content entities such as image, video, audio, collections of multimedia document and so on. Figure 6.3 shows the various types of content description tasks. The light gray elements of Figure 6.3 represent description schemes that are discussed in other chapters of this book. In between parenthesis, the chapter number and the type (if different from the element name) are indicated. For instance, the element *Image* is of type *StillRegion* and is discussed in Chapter 8 [7].

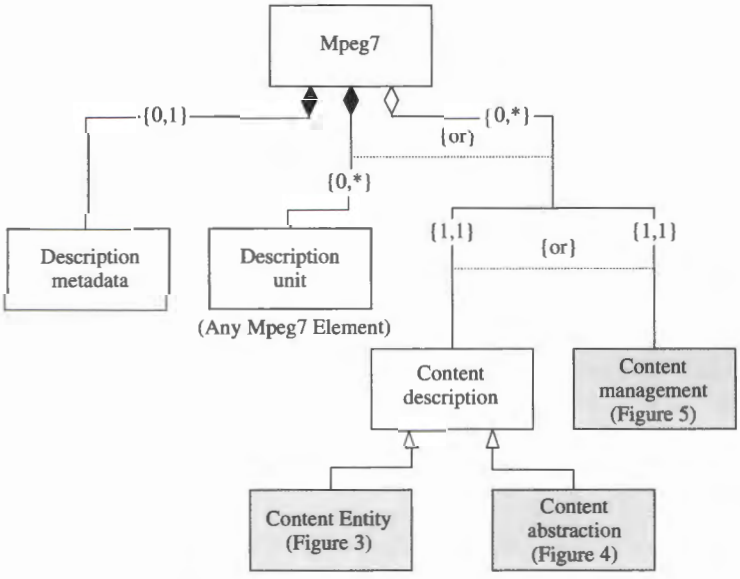


Figure 6.2 Organization of *Mpeg7* root and top-level elements

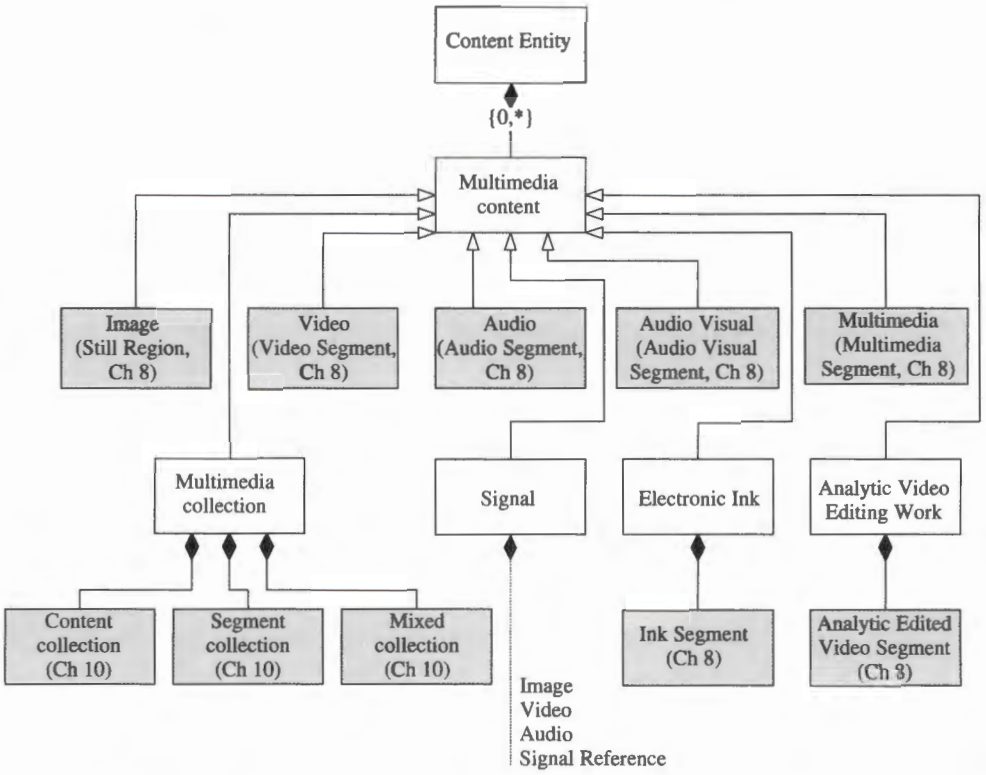


Figure 6.3 Organization of *ContentEntity* elements

As an example, the model for describing an image is presented in Figure 6.4. In MPEG-7, images can be described as a specific Segment called StillRegion (see Chapter 8 [7] for more details). The description of the image relies on description schemes and descriptors addressing the following features: Header, Media information, Media locators (to localize the content), Creation information, Usage information, Spatial locators, Spatial masks (do localize the shape and possibly its various connected components), visual features (visual descriptors, description schemes and GridLayout), textual annotation and specific features such as Structural unit, Matching hint, Point of view and Relation. Note that many of these elements are optional. The image can be described by Visual description schemes and descriptors that address low-level features such as color, texture and so on. Finally, the image can be segmented into subimages by spatial decomposition. The subimages are themselves StillRegions that can be described essentially as images. Many of the features involved in this example are more precisely described in Chapters 7–11 [6–10].

- *Description of ContentAbstraction*: These elements allow the description of summaries of multimedia content, different views of the image, video or audio signals, variations of the content, models of the content and real-world semantics related to multimedia content. Figure 6.5 shows the organization of the ContentAbstraction elements. As can be seen, there are five major abstraction tasks that use specific description schemes

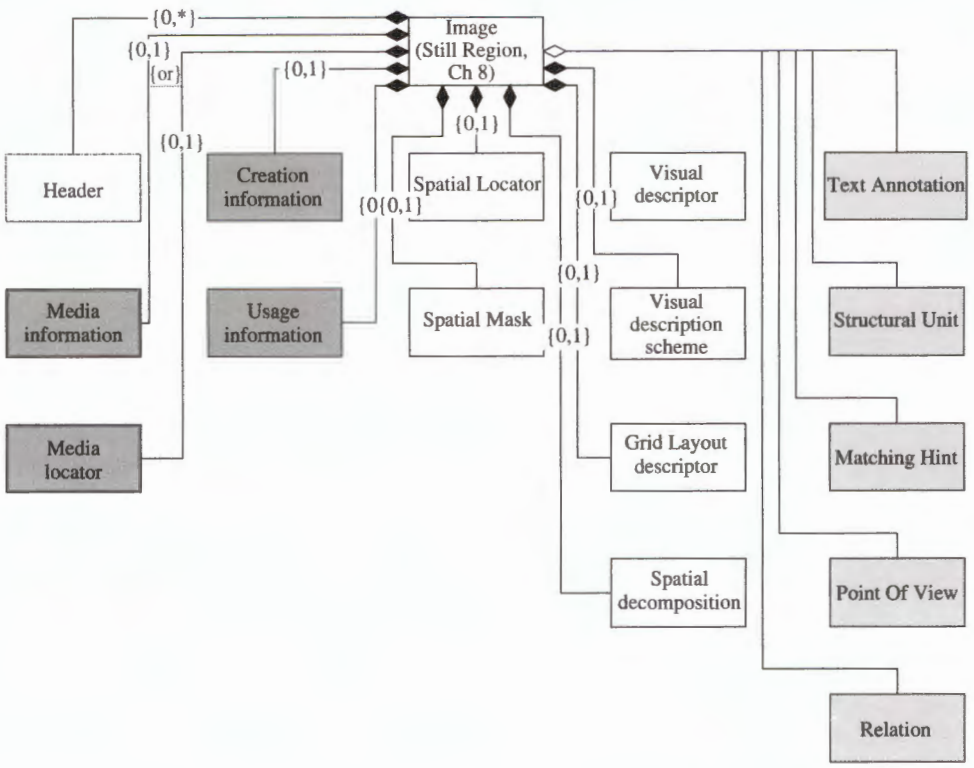


Figure 6.4 Example of the *Image* top-level element

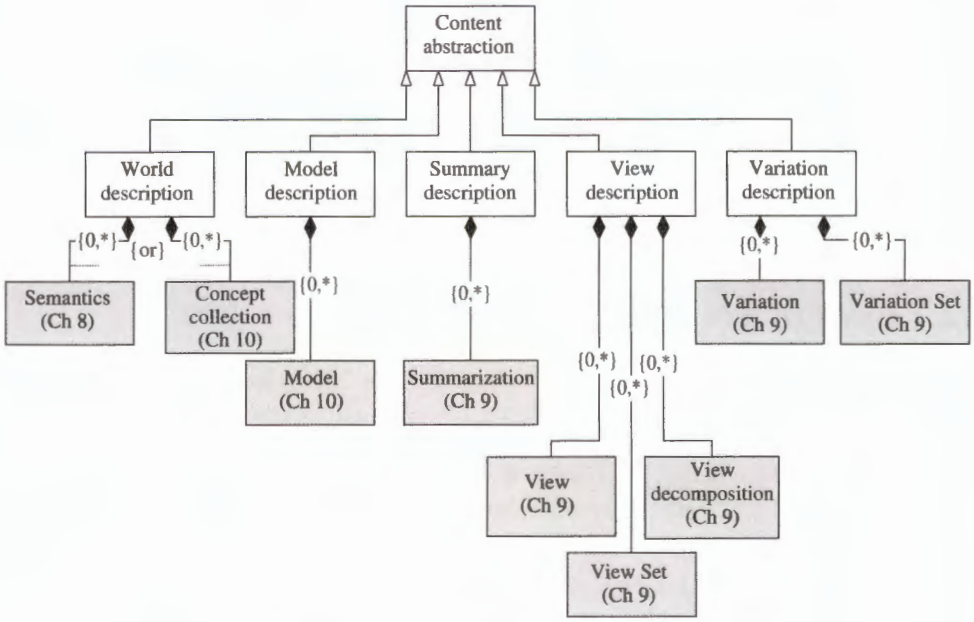


Figure 6.5 Organization of *ContentAbstraction* elements

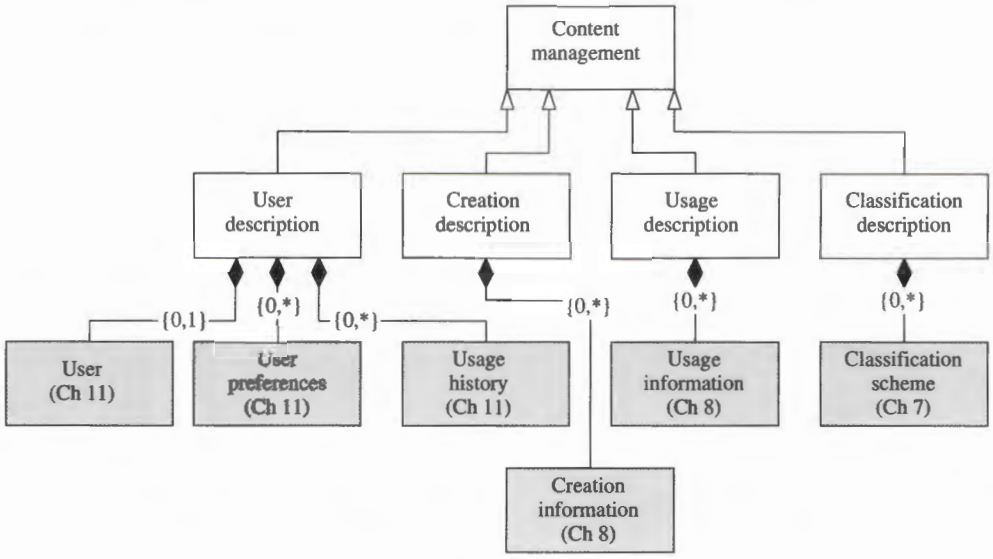


Figure 6.6 Organization of *ContentManagement* elements

(light gray elements of Figure 6.5), which are presented in Chapters 8 [7], 9 [8] and 10 [9].

- *ContentManagement*: This element is illustrated on Figure 6.6. It defines the appropriate set of tools to perform the most usual management tasks. This includes the description of the creation, the classification, the usage of the multimedia content as well as the user consuming the multimedia content. More information about the User, the UserPreferences, and the UsageHistory can be found in Chapter 11 [10]. Creation and Usage information are described in Chapter 8 [7]. Finally, ClassificationSchemes are discussed in Chapter 7 [6].

As can be seen, the *Mpeg7* root element and the Top-level elements are used to define a model for valid description. They also represent a selection of a certain number of descriptors and description schemes from the MPEG-7 standard to efficiently perform the most usual content description and management tasks.

6.4 CONCLUSION

In this chapter, we presented an overview of the MDSs developed by MPEG as part of the MPEG-7 standard. The MDSs are metadata structures for describing and annotating multimedia content, and provide a way to describe in XML the important concepts related to multimedia content. The objective is to allow interoperable searching, indexing, filtering and access by enabling interoperability among devices that deal with multimedia content description.

We presented the organization of the MPEG-7 MDSs into different areas of Basic Elements, Content Description, Content Management, Content Organization, Navigation and Access and User Interaction. In the process of developing the MPEG-7 MDSs, efforts

have been made to harmonize with other multimedia metadata standards and existing practices. Beyond the standardized set of MPEG-7 MDSs, the MPEG-7 DDL also allows the extension of the MPEG-7 standard to accommodate specific multimedia content domains and applications.

ACKNOWLEDGMENT

The MPEG MDSs presented in this chapter result from the contributions and collaborative efforts of many people. The authors are particularly grateful to the other editors of the MDS specification including P. van Beek, A. Benitez, J. Heuer, J. Martinez, Y. Shibata and T. Walker. The authors also thank all members of the MPEG MDS Group for their many contributions toward the development of the MPEG-7 standard.

REFERENCES[2]

[1] MPEG Requirements Group, "MPEG-7 requirements", Doc. ISO/MPEG N4320, Sydney MPEG Meeting, July 2001.
[2] S. S. Alhir, *UML in a Nutshell*, O'Reilly & Associates, Sebastopol, Calif., 1998.
[3] M. Wollborn, C. Thiénot and J. Heuer, BiM – The Binary Format for MPEG-7 Data, Chapter 5.
[4] J. Hunter and C. Seyrat, MPEG-7 Description Definition Language, Chapter 4.
[5] XML Schema Part 0: Primer, Part 1: Structures, Part 2: Datatypes, W3C Proposed Recommendation, 16 March 2001. *http://www.w3.org/TR/*.
[6] T. Walker, J. Heuer and J. M. Martinez, MPEG-7 Basic MDS elements, Chapter 7.
[7] A. B. Benítez, J. M. Martínez, H. Rising and P. Salembier, Description of Audio-Visual Documents, Chapter 8.
[8] P. van Beek and J. Smith, Navigation and Access, Chapter 9.
[9] A. B. Benítez and J. Smith, Description of Collection of Audio-Visual content and Models, Chapter 10.
[10] P. van Beek, K. Yoon and M. Ferman, User Interaction, Chapter 11.
[11] MPEG MDS Group, "MPEG-7 Multimedia Description Schemes XM (v8.0)", Doc. ISO/MPEGN4243, Singapore MPEG Meeting, March 2001.
[12] ISO/IEC 15938-5:2001, "Multimedia Content Description Interface – Part 5 Multimedia Description Schemes," version 1.

[2] Although MPEG documents are in principle nonpublic documents, many of them are made public and can be accessed at the MPEG Home Page, *http://mpeg.telecomitalialab.com/*. Nonpublic MPEG documents may be obtained through the MPEG Head of Delegation of the respective country.

7

Basic Elements

Toby Walker[1], Jörg Heuer[2] and José M. Martínez[3]

[1]*Sony Corp., Tokyo, Japan,* [2]*Siemens AG, Munich, Germany,* [3]*Polytechnic University of Madrid, Spain*

7.1 INTRODUCTION

At the heart of an MPEG-7 multimedia description is a set of description tools called the MPEG-7 *basic elements*. The basic elements deal with the essentials of multimedia content description, including time, linking and media localization, annotation, and defining description vocabularies. Unlike the other parts of MPEG-7, which are organized around different aspects of multimedia content description, the basic elements are a library of common description tools. They are used in the Audio, Visual and Multimedia Description Schemes (MDS) parts of MPEG-7.

In this chapter, we present an overview of the MPEG-7 basic tools, explain the role each tool plays in MPEG-7 and demonstrate the use of these tools with examples. A summary of the tools we cover in this chapter is shown in Figure 7.1. First, we introduce the *basic data types*, which represent mathematical constructs useful for multimedia description, such as matrices and vectors. Next we describe the *linking and localization tools*, which are used to specify references within descriptions, to link MPEG-7 descriptions to media, to identify and locate media and to describe time. Then we introduce the MPEG-7 basic tools that address common aspects of multimedia content description. This includes graphs for structuring complex multimedia content descriptions, text annotations, descriptions of people and places, specifications of affective response and description ordering. In addition, the basic tools have facilities for defining and referencing vocabularies for multimedia content descriptions using classifications schemes and terms. The *schema tools*, which are greyed out in Figure 7.1, are covered separately in Chapter 6 [1].

In this chapter, we will be illustrating the use of the basic elements through examples based on a goal scene taken from the video of a soccer game, shown in Figure 7.2.

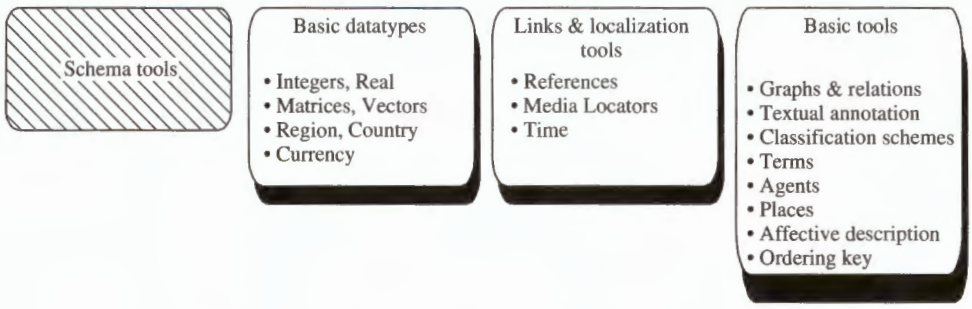


Figure 7.1 Overview of the MPEG-7 basic elements



Figure 7.2 Spain scores a goal against Sweden (MPEG-7 content set, courtesy Televisión Española RTVE)

7.2 BASIC DATA TYPES

This section presents a brief summary of the MPEG-7 basic data types. While the MPEG-7 DDL already provides a library of common data types (integers, floating-point numbers, etc.), multimedia content description needs some additional mathematical constructs. Therefore, MPEG-7 defines a set of data types useful for multimedia content description, which are summarized in Table 7.1.

Of particular importance to multimedia content description are the vector and matrix data types, which represent vectors and n-dimensional matrices. The vector and matrix data types are used extensively in both the audio and visual description tools and in the MPEG-7 content models Chapter 10 [2].

Table 7.1 Basic data types defined in MPEG-7

Data types	Represent
Integers and reals	Constrained integer and real values.
Vectors and matrices	Arbitrary sized vectors and matrices of integer or real values.
Probability vectors and matrices	Probability distributions described using vectors and matrices.
Strings	Codes identifying content type, countries, regions, currencies and character sets.

7.3 LINKING AND LOCALIZATION TOOLS

In this section, we introduce the MPEG-7 description tools for linking and localization, which are tools for referencing MPEG-7 descriptions, for linking those descriptions to multimedia content and for describing time in multimedia content.

References

We often need to refer from one description to another description without being constrained by the Extensible Markup Language (XML) hierarchical document structure or by document boundaries. MPEG-7 provides a single powerful tool, the Reference data type, for referring to any description element (i.e. a descriptor, description scheme or their subcomponents). The Reference data type provides three basic reference mechanisms:

- *idref*: References a description element within the same description document. The target is identified by its ID attribute, which is unique within the description document.
- *xpath*: References a description element using a subset of XML Path Language (*XPath*) [3, 4]. An XPath expression identifies a reference target by its position within the description tree.
- *href*: References a description or description element using a Uniform Resource Identifier (URI). Unlike the idref and Xpath mechanisms, href references can refer to an element in another description document.

This data type is used in the Relation description scheme (see Section 7.4) and the Affective description scheme (see Section 7.4).

In addition to the Reference data type, two data types for representing xPath expressions for *selectors* and *fields* are defined: xPathSelector and xPathField. The xPathSelector data type selects a set of elements in a description while the xPathField data type picks out one element from a description. These data types are used in the OrderingKey description scheme (see Section 7.4).

Identifying and locating media

MPEG-7 provides two ways to link to media: *implicitly* by identifying the media with a unique identifier and *explicitly* by pointing to the media with a media locator.

Unique identifiers

The UniqueID data type is used to uniquely identify an item, such as multimedia content or an MPEG-7 description. It can accommodate a variety of identifiers such as URIs, International Standard Work Code (ISWC) or application-specific identifiers. If required, each UniqueID can include attributes that specify the type of identifier (e.g. ISWC), the organization defining that identifier type (e.g. ISO) and the authority that assigned the identifier.

Media locators

The media locator data types link a description element containing the media locator to multimedia content. There are three kinds of media locators: a generic MediaLocator, a TemporalSegmentLocator and an ImageLocator.

A MediaLocator can link to multimedia content by using a URI to refer to the multimedia content or by including the multimedia content inline. Unlike the UniqueID, the URI in a MediaLocator physically locates multimedia content for access and is not an identifier. Additionally, when the multimedia content contains multiple streams (e.g. two or more audio streams in a video), a streamID can be used to select a stream within the multimedia content.

In addition to the generic localization mechanisms of a MediaLocator, two other types of media locators are defined. One, TemporalSegmentLocator, locates a segment within temporal multimedia, for example, audio or video. The time period of the segment can be specified using the media time data types or using the offset and length in bytes of the segment within the media. Another, ImageLocator, locates an image or a single frame in a video. It is restricted to media time points instead of media time periods.

Suppose that we want to locate a video segment in the soccer video shown in Figure 7.2, which runs for the first 3 to 13 seconds of the video, then we could use the following TemporalSegmentLocator:

```
<TemporalSegmentLocator>
    <MediaUri>file:soccer.mpg</MediaUri>
    <MediaRelTimePoint timeBase="../MediaUri">
        PT3S
    </MediaRelTimePoint>
    <MediaDuration>PT10S</MediaDuration>
</TemporalSegmentLocator>
```

If, on the other hand, we want to describe a soccer team logo then the inline format of ImageLocator is useful as it allows us to include the image data right inside the description itself. This is shown in the example below (the media data has been omitted):

```
<ImageLocator>
    <InlineMedia type="image/jpeg">
        <MediaData64>/9j/4AAQSkZJRgABAQEAgAAD...</MediaData64>
    </InlineMedia>
</ImageLocator>
```

Time

MPEG-7 can represent two different kinds of time: (1) *media time*, which is time measured or stored within the media and (2) *generic* or *world time*, which is time measured in the world. Both media and world time use the same representation, except that the data types for world time also contain time zone (TZ) information. Here we describe only the media time data types.

The MPEG-7 media time data types are compatible with time specifications used in common multimedia formats such as MPEG-4 [5] or Synchronized Multimedia Integration Language (SMIL) [6] and are based on the ISO 8601 standard [7]. The media time data types represent *time periods* using a *start time point* (mediaTimePoint data type) and a *duration* (mediaDuration data type). The mediaTimePoint data type uses the following syntax:

```
-YYYY-MM-DDThh:mm:ss:nFN
```



which includes the year (Y, which can be a negative value for B.C. dates), months (M), days (D), a separator T, hours (h), minutes (m) and seconds (s), 1/N is a fraction of one second and *n* the number of those fractions. Fractions are used for subsecond times instead of the ISO8601 decimal form in order to ensure compatibility with some multimedia standards. The beginning of the soccer broadcast (see Figure 7.2) on March 25, 1998 at 9 P.M., three minutes, thirteen seconds and 10/30 of a second can be described by:

```
<MediaTimePoint>1998-03-25T21:03:13:10F30</MediaTimePoint>
```

The mediaDuration data type uses the following format:

```
(-)PnDTnHnMnSnNnF
```

In this format, each part of the duration contains a count followed by a letter indicating the unit being counted: P is the separator indicating the start of a duration, days are indicated by D, T separates time from days, H indicates hours, M minutes, S seconds and the subsecond part uses the same representation as mediaTimePoint, with N indicating the counted fractions and F the fractions of one second. For example, a play in a soccer game that lasts 2 minutes, 10 seconds and 25/30 of a second in a soccer game can be specified by:

```
<MediaDuration>PT2M10S25N30F</MediaDuration>
```

On top of the mediaDuration and mediaTimePoint data types, MPEG-7 builds three kinds of media time representation:

- *Simple time*: (Figure 7.3a) The basic representation of an absolute time point.
- *Relative time*: (Figure 7.3b) Specifies a media time point relative to a time base. Useful if a media segment, such as a story in a news sequence, is placed dynamically. To update the story's description, only the time base (t_0) needs to be changed.
- *Incremental time*: (Figure 7.3c) Specifies a time period by counting predefined time units. When dealing with sampled data, the sample points or ticks of the reference clock are often equidistant in time.

For example, suppose we want to describe the start frame for a play in the soccer video, we can use a relative incremental time point to do so:

```
<MediaRelIncrTimePoint timeUnit="PT1N30F"
      timeBase="../../MediaTimePoint[1]">
  45897
</MediaRelIncrTimePoint>
```

The start time is described relative to the beginning of the broadcast, indicated by timeBase referring to the Media Time-Point specification in the grandparent of the current node (not shown). The value for timeUnit (i.e. frame rate) indicates that units of 1/30th of a second are counted. Finally, the time is 45 897 time units (i.e. frames) from the start of the video.

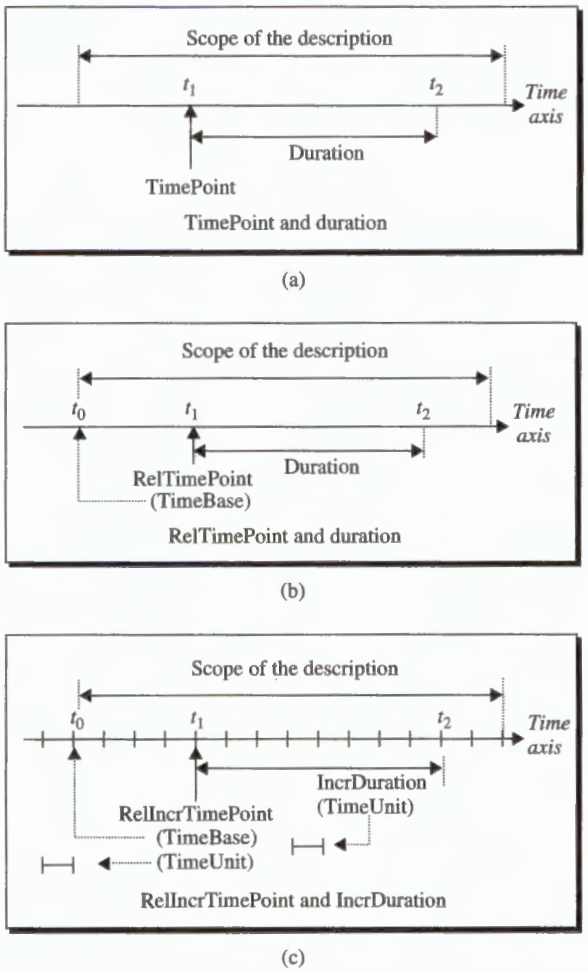


Figure 7.3 Time representations (a) basic time specification with global scope; (b) relative time point specification with respect to a time base and (c) incremental time specification

7.4 BASIC TOOLS

This section presents an overview of the MPEG-7 basic description tools. The basic description tools are a library of description schemes and data types, which are used as primitive components for building the more complex and functionality-specific description tools found in the rest of MPEG-7.

In this section we cover the *graph* and *relation* tools, which weave together complex multimedia description structures; *textual annotations*, which represent textual descriptions; *classification schemes and terms*, which define and reference vocabularies for multimedia content descriptions; *people and places*, which represent the people and places related to multimedia content, and *affective description*, which describes emotional response to multimedia content and finally, the *ordering tools*, which describe different ways to order descriptions for presentation.

7.4.1 Relations and Graphs

The basic structure of an MPEG-7 multimedia content description follows the hierarchical structure provided by nested elements in an XML document. However, some multimedia content descriptions are best represented as graphs with different description scheme instances linked together with relations, for example, the description of a narrative or the spatial configuration of entities in an image. MPEG-7 supports this with a Relation description scheme for representing relations between description schemes and a Graph description scheme for representing graph-structured descriptions.

Relations

The Relation description scheme represents a relation between one or more description schemes instances. In MPEG-7, all relations are directed and the arguments of each relation are divided into *source* and *target* arguments, represented by the values of the source and target attributes, respectively. Finally, an optional strength value represents 'fuzzy' relations.

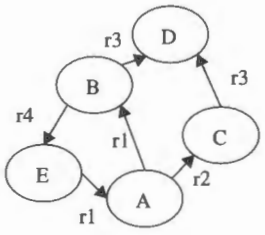
In MPEG-7, the type of a relation is defined as a term in classification schemes. The Relation description scheme identifies the type of the relation using a type attribute, which references a term defined in a classification scheme. Some classification schemes for relations are normative, for example, the segment relations [8], while others will be defined by applications using MPEG-7.

The Relation description scheme supports two formats for relations: (1) an *external* form in which the relation is written apart from the descriptions of the description schemes instances being related and (2) an *internal* form that embeds the relation directly within the description of the source argument of the relation.

Graphs

The Graph data type represents a graph as a list of nodes and relations between those nodes, written in any order. We demonstrate how a simple graph can be written using the Graph description scheme and the Relation description scheme. Figure 7.4 shows a graph and its corresponding MPEG-7 description.

For each node in the graph, there is a corresponding Node element that is identified by its id attribute. For each relation in the graph, a corresponding Relation element specifies the type of the relation and the source and target attributes indicate the nodes joined by relation. The value for both attributes is a URI reference to the node's identifier.




```
<Graph>
    <Node id="A"/> <Node id="B"/>
    <Node id="C"/> <Node id="D"/>
    <Node id="E"/>
    <Relation type="#r1" source="#A" target="#B"/>
    <Relation type="#r2" source="#A" target="#C"/>
    <Relation type="#r3" source="#B" target="#D"/>
    <Relation type="#r3" source="#C" target="#D"/>
    <Relation type="#r4" source="#B" target="#E"/>
    <Relation type="#r1" source="#E" target="#A"/>
</Graph>
```

Figure 7.4 Graph description: a simple graph and its MPEG-7 representation

For further examples of how the Graph and Relation description schemes are used in MPEG-7 for multimedia content description, see the use of graphs for structural and semantic description in [8].

7.4.2 Text Annotation

The description of multimedia content using natural language text is called text annotation. The practice of using text annotations to search, catalogue and index multimedia content is both longstanding and widespread. MPEG-7 supports this practice with the TextAnnotation data type that allows *free text*, *keyword*, *structured* and *dependency structure* annotations. In the rest of this section, we show how each of these formats can be used to annotate the soccer goal scene shown in Figure 7.2.

Free text

Suppose that an annotator has created an annotation of the goal scene: 'Spain scores a goal against Sweden. The scoring player is Morientes.' We can create an instance of TextAnnotation data type using free text:

```
<TextAnnotation>
    <FreeTextAnnotation xml:lang="en">
       Spain scores a goal against Sweden.
       The scoring player is Morientes.
    </FreeTextAnnotation>
</TextAnnotation>
```

The annotation itself appears unmodified in the FreeTextAnnotation element with the xml:lang attribute indicating that the text is written in English.

Keyword annotations

While a free text annotation is easily understood by a human reader, it is difficult for a machine to process natural language text. Instead of free text annotation, we can use a KeywordAnnotation that distils the annotation into a set of keywords:

```
<TextAnnotation>
    <KeywordAnnotation>
          <Keyword>score</Keyword>
          <Keyword>Sweden</Keyword>
          <Keyword>Spain</Keyword>
          <Keyword>Morientes</Keyword>
    </KeywordAnnotation>
</TextAnnotation>
```

For a computer, keywords are much easier to process than unrestricted natural language. Notice that in this example we have extracted the most important words and phrases from the free text. However, in flattening an annotation into a list of keywords we have lost the structure in the original text. For example, from the keywords we cannot determine who scored the goal. As we see next, the Structured Annotation data type adds back some of this structure while being more amenable to computer processing than free text.

Structured annotations

Suppose we annotate the soccer goal shown above. One natural way to go about is to answer a set of questions: Q: What happened? A: A goal was scored. Q: Who did it? A: Spain. Q: When? A: March 25, 1998. Q: Where? A: A Coruña, Spain. A StructuredAnnotation is built around such questions:

```
<TextAnnotation>
   <StructuredAnnotation>
         <Who><Name>Spain</Name></Who>
         <WhatAction><Name>score goal</Name></WhatAction>
         <Where><Name>A Coruña, Spain</Name></Where>
         <When><Name>March 25, 1998<Name></When>
   </StructuredAnnotation>
</TextAnnotation>
```

The answer to each question is represented by a single element: Who, WhatObject, WhatAction, Where, When, Why and How. The content of each of these elements is of type TermUse.

StructuredAnnotation strikes a balance between simplicity and expressiveness. While it is a simple format that captures some of the structure of an annotation, it is rich enough to capture the basic semantic structure of an annotation.

Dependency structure

The DependencyStructure data type provides a powerful description tool when we want to represent the linguistic structure of an annotation. It is based on a linguistic theory called *dependency grammar* [9, 10] that explains a sentence's grammatical structure in terms of dependencies between its elements. The structure is represented using a dependency tree as shown in Figure 7.5. The tree consists of subtrees; each of which contains a *governor* and a set of *dependents*. In Figure 7.5, the dependents at the end of the arrows depend on the governor at the beginning of the arrow. The governor determines the grammatical properties of the phrase that consists of the governor itself and its dependents.

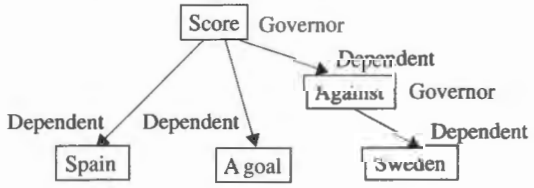


Figure 7.5 Dependency structure for 'Spain scored a goal against Sweden'

The DependencyStructure data type maps the dependency structure of a sentence into a nested set of XML elements. A Phrase element represents one subtree of the dependency tree and contains one Head and any number of child Phrases. The Head is the governor of its parent Phrase and each child Phrase represents a subtree that depends on that Head. For example, the annotation 'Spain scored a goal against Sweden' is represented as follows:

```
<TextAnnotation>
    <DependencyStructure>
          <Sentence>
                <Phrase operator="subject">
                      <Head type="noun">Spain</Head>
                </Phrase>
                <Head type="verb" baseForm="score">
                     scored
                </Head>
                <Phrase operator="object">
                      <Head type="article noun">a goal</Head>
                </Phrase>
                <Phrase>
                      <Head type="preposition">against</Head>
                      <Phrase>
                            <Head type="noun">Sweden</Head>
                      </Phrase>
                </Phrase>
          </Sentence>
     </DependencyStructure>
</TextAnnotation>
```

At the sentence level, the verb 'score' is the head with three dependent phrases for its subject, object and 'against Sweden'. The operator attribute indicates the type of dependency relation; for example, that 'Spain' is the subject and 'goal' is the object.

Using the DependencyStructure data type, we can match annotations using the syntactic structure of an annotation. For example, we could search for all annotations in which the subject was 'Spain' and the verb was 'score'. This allows us to find all cases where Spain scores against someone, which is difficult using only unstructured text matching.

Classification schemes and terms

The MPEG-7 description schemes and Descriptors provide a framework for multimedia content description. However, the vocabulary used inside a description is of equal importance. To interoperate, we need to standardize those vocabularies as well. Moreover, MPEG-7 applications need the flexibility to define their own vocabularies and extend existing vocabularies. However, it is not possible for all such vocabularies to be standardized within MPEG-7. To meet these requirements, MPEG-7 provides *classification schemes* to define a vocabulary for describing a domain as a set of terms. An MPEG-7 description can then reference the terms defined by a classification scheme within a multimedia content description. MPEG-7 includes, for example, informative classification schemes that define sets of terms for describing genre and multimedia formats.

Defining classification schemes

The ClassificationScheme description scheme contains a list of term definitions and, optionally, imported classification schemes. Each ClassificationScheme instance is uniquely identified by a URI. An optional domain attribute indicates the set of descriptors to which the classification scheme can be applied. This allows classification schemes providers to specify hints for using their vocabularies within MPEG-7 descriptions.

The terms in a classification scheme are described using TermDefinitions, which contain a term identifier, one or more names (allowing multilingual classification schemes)

and definitions. Each term can contain related terms to create a hierarchy of terms via relations such as broader term/narrower term.

The examples below show part of a ClassificationScheme for sports:

```
<ClassificationScheme uri="urn:examples:cs:SportsCS">
    <Term termID="soccer">
          <Name xml:lang="en">Soccer</Name>
          <Name xml:lang="es">Fútbol</Name>
    </Term>
    <Term termID="basketball">
          <Name xml:lang="en">Basketball</Name>
          <Name xml:lang="es">Baloncesto</Name>
    </Term>
    <Term termID="baseball">
          <Name xml:lang="en">Baseball</Name>
          <Name xml:lang="es">Béisbol</Name>
    </Term>
    <!-- Other sports terms -->
</ClassificationScheme>
```

This classification scheme is identified as 'urn:examples:cs:SportsCS' by its uri attribute. Each term in this classification scheme represents a different kind of sports such as soccer and so on. While each term has names in both English and Spanish, the identifier for each term (e.g. termID) is language independent.

Using terms

There are two data types for specifying a term in a description: TermUse and ControlledTermUse. The TermUse data type contains an optional term reference and an optional inline term definition, allowing three different uses of a term in a description:

- *Term reference use*: References a term in a classification scheme. Only the term's URI is included (via the href attribute). No term definition is included in the TermUse instance.
- *Inline term use*: References a term in a classification scheme (href) and also includes an inline term definition (name, definition and related terms). This allows writing self-contained descriptions.
- *Free term use*: The term is defined directly in the instance and is not related to any classification scheme. No term reference (href) is present in this ad hoc term, only the term definition (name, definition and related terms).

The ControlledTermUse data type is the same as TermUse except that it makes the term reference mandatory and therefore does not support the free term use.

There are two forms for referring to a term: using URIs and an abbreviated form based on classification schemes aliases. A URI term reference (usually a Universal Resource Name) uniquely identifies a term. However, an abbreviated alias term reference includes only the classification scheme alias and the termID.

Suppose we want to reference the 'soccer' category defined earlier in the sports classification scheme, we can use the following form:

```
<TermUse href="urn:examples:cs:SportsCS:soccer"/>
```

In the example, the href attribute references the term defined in the classification scheme. The first part, 'urn:examples:cs:SportsCS' is the identifier for the classification scheme and matches the value given in the uri attribute for the ClassificationScheme instance. The second part of the href value, 'soccer' identifies the term and matches the termID within the ClassificationScheme.

We can also use the inline form to include part of the term's definition inside the TermUse:

```
<TermUse href="urn:examples:cs:SportsCS:soccer">
       <Name xml:lang="en">Soccer</Name>
</TermUse>
```

Finally, we can even specify a free term that represents an ad hoc term and makes no reference to a classification scheme:

```
<TermUse><Name xml:lang="en">Mai lai</Name></Genre>
```

It is the lack of an href attribute that tells us that this is a free term.

Graphical classification schemes

Graphical classification schemes allow classification schemes to be used for graphs. There are several uses for classifications and thesauri of graphs. The first is storing description templates to be instantiated in a description; for example, templates of commonly used semantic descriptions [8]. Second, validating description instances that should have a particular application defined structure. Third, providing a mechanism for graph production rules, which allow graphically structured descriptions to be restricted or generalized according to a set of rules expressed as graph transformations.

People and places

In this section, we introduce the MPEG-7 description tools for describing people and places.

1. Agents: In MPEG-7, persons (e.g. actor, director), organizations (e.g. a company) and groups of persons (e.g. a musical ensemble) are all treated as *agents*, An agent which is an entity responsible for an action.

The Person description scheme describes a real or fictional person. It includes the name (given, family, title etc.), the affiliation of the person to an organization or a group and the address (e.g. telephone, fax, e-mail, URL).

The PersonGroup description scheme describes a group of persons having a collective title and whose member persons are easily identified (i.e. a musical group, a basketball team or a dance ensemble). It includes the group's name and, optionally, the kind of group (e.g. a rock band), the members of the group and the jurisdiction and the address of the group.

The Organization description scheme describes an organization: a group of persons with a collective title whose member persons are not easily identified or whose membership is very large, that is, professional associations, production companies and so on. It includes the organization's name and, optionally, the kind, contact, jurisdiction and address of the organization.

The example below shows an excerpt of a description of the Spanish National Soccer Team that played in the soccer game shown in Figure 7.2.

```
<PersonGroup>
    <Name>Spanish National Soccer Team</Name>
    <Kind><Name>Soccer Team</Name></Kind>
    <Member xsi:type="PersonType">
          <GivenName>Fernando</GivenName>
          <FamilyName>Morientes</FamilyName>
    </Member>
    :
</PersonGroup>
```

2. Places: The Place description scheme describes existing, historical and fictional places. It is used in MPEG-7 for describing places depicted in the content (e.g. historical and fictional settings, locations of events) and places in the real world (e.g. shooting location). It can include the name, role, geographic position (longitude, latitude and altitude), country, region and postal address.

The example below describes the location of the stadium shown in Figure 7.2.

```
<Place>
    <Name xml:lang="es">Estadio de Riazor</Name>
    <Name xml:lang="en">Riazor Stadium</Name>
    <Role><Name>soccer match location</Name></Role>
    <Region>es</Region>
</Place>
```

Affective response

The Affective description scheme characterizes a viewer's affective – interpretative, emotional or mood – response to multimedia content. It assigns a numeric score to each member of a set of descriptions (e.g. the segments in a video). Each score measures the relative (within the group of instances) intensity of the affective response on a scale, such as excitement, happiness, anger and so forth.

For example, suppose we record one person's excitement level while viewing a video of the soccer match in Figure 7.2. We might obtain the following affective response description:

```
<Affective>
    <Type><Name>Excitement</Name></Type>
    <Score idref="Seg0">0.00360117</Score>
    <Score idref="Seg2">0.825006</Score>
    <Score idref="Seg3">-0.20360117</Score>
    :
</Affective>
```

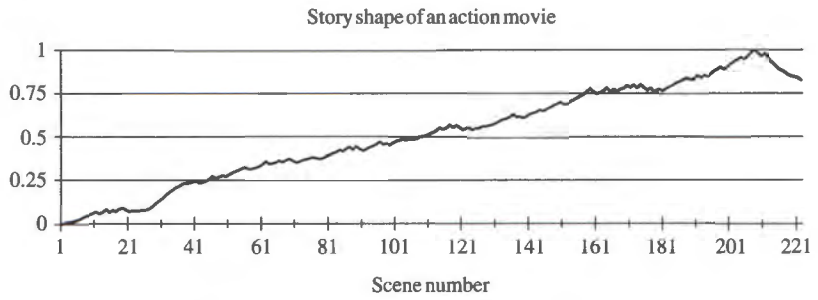


Figure 7.6 Story shape of an action movie

In this example, the Type element indicates that the affective scale being measured is 'Excitement'. This is then followed by a set of affective scores represented as Score elements, each of which is a value between −1 and 1. Each Score associates an excitement value with a video segment, using the idref attribute. For space reasons, the video segments are not shown here. In this example, large negative values indicate 'boring' segments and large positive values 'exciting' segments.

Another application of the Affective description scheme is to represent interpretative information. For example, suppose we ask an audience to indicate their interpretation of the story complication (i.e. the tension level) while watching an action movie. The resulting values when plotted against time capture the 'shape' of the story's development. For example, Figure 7.6 shows the affective scores for scenes (segments) in an action movie. The affective score ascends toward the climax, which appears as a peak in the graph.

Ordering descriptions

When accessing large sets of data, one of the most common operations we use to organize the data is ordering data based on some key. The OrderingKey description scheme provides a hint for ordering descriptions for presentation, using information contained within those descriptions.

For example, we can order a set of video segments in a soccer game by the amount of camera zoom (which can suggest events of interest) within each segment:

```
<OrderingKey direction="descending">
      <Selector xpath="TemporalDecomposition/AudioVisualSegment"/>
      <Field
xpath="CameraMotion/CameraMotionSegment/FractionalPresence@ZOOM_IN"/>
</OrderingKey>
```

The OrderingKey contains a Selector element that identifies the description items to be ordered and one or more Field elements that form a key for ordering. Both the Selector and Field elements are specified using XPath [3].

REFERENCES

[1] P. Salembier and J. Smith, Overview MPEG-7 Multimedia Description Schemes, Chapter 6.
[2] J. Smith and A. B. Benítez, Content Organization, Chapter 10.
[3] XML Path Language (XPATH), Version 1, W3C Recommendation, 16 November 1999, *http://www.w3.org/TR/1999/REC-xpath-19991116*.
[4] ISO/IEC 15938-5:2001, Multimedia Content Description Interface – Part 5 Multimedia Description Schemes, version 1.
[5] ISO/IEC 14496-1, Coding of audio-visual objects – Part 1: Systems, 1999.
[6] W3C Working Draft. Synchronized Multimedia Integration Language (SMIL 2.0), March 2001, *http://www.w3.org/TR/smil20*.
[7] ISO 8601, Representations of dates and times, 1988-06-15.
[8] A. B. Benítez, J. M. Martínez, H. Rising, P. Salembier, Description of Audio-Visual documents, Chapter 8.
[9] I. Mel'cuk, *Dependency Syntax: Theory and Practice*, State University of New York Press, New York 1988.
[10] R. Hudson, "Recent Trends in Dependency Syntax", *Handbücher zur Sprach- und Kommunikationswissenschaft*, de Gruyter, Berlin, 1990, pp. 329–338.

8

Description of a Single Multimedia Document

Ana B. Benitez[1], José M. Martinez[2], Hawley Rising[3] and Philippe Salembier[4]

[1]Columbia University, New York, USA, [2]Universidad Politècnica de Madrid, Madrid, Spain, [3]Sony Network and Software Solutions Center of America, San Jose, California, USA, [4]Universitat Politècnica de Catalunya, Barcelona, Spain

8.1 INTRODUCTION

MPEG-7 provides description tools for describing one multimedia document. Multimedia documents supported by MPEG-7 are not only typical audiovisual documents such as image, videos, audio sequences and audiovisual sequences but also special types of multimedia documents such as ink content, MPEG-4 presentations and Web pages with different media (e.g. images and text). These tools are applicable to multimedia content of any format including digital and analog multimedia content.

In this chapter, we present the MPEG-7 tools that describe the management, structure and semantics of a single multimedia document. The management tools describe the creation, media and usage of the content. The structural tools describe spatial and/or temporal segments of multimedia content, their attributes and the relationships among them. Examples of segments are still regions in images and a set of frames in videos. The semantic tools describe semantic entities in narrative worlds depicted by multimedia content together with their attributes and the relationships among them. Examples of semantic entities are objects, events and concepts. The tools described in this chapter use tools that are described in other chapters of the book such as audio description tools, video description tools, basic description tools [8] and collection description tools [2]. Some of these tools can also describe certain aspects of multimedia collections such as the creation date, the media format and rights information, among others [2].

An example of a content management, structure and semantic description of an image is illustrated by Figure 8.1. In this example, the creation information tools specify the photographer who took the picture as well as the place and time where and when

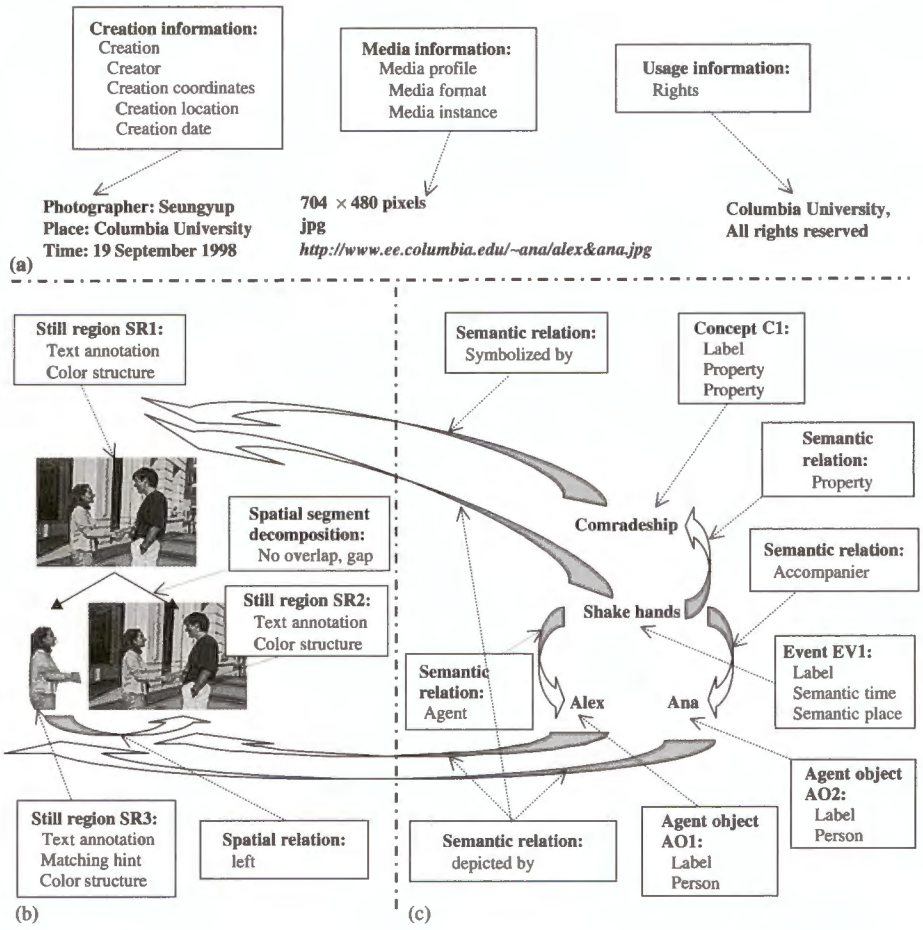


Figure 8.1 Example of a content management: (a) structure; (b) semantic; and (c) description of an image

the photograph was taken, respectively. The size, coding format and locator of the image is described using the media information tools. Rights information is represented by the usage information tools. The image and two regions of the image are described using still region tools. The decomposition of the image into the two regions and a spatial relationship between these two regions are captured using segment decomposition tools and structural relation tools, respectively. In terms of semantics, two persons, an event and a concept in the image are described using semantic entity tools. Several relationships among these semantic entities are characterized using semantic relation tools. The description in Extensible Markup Language (XML) format for this and the other examples in this chapter is included in the corresponding appendix stored on the DVD accompanying this book.

8.2 CONTENT MANAGEMENT

The content management tools allow the description of the life cycle of multimedia content from its creation to its usage through encoding. They are organized in three

main areas dealing with the description of the media, of the creation process and of the usage. The media information describes storage format, coding parameters, media quality, media location, and so on. The creation information gathers author-generated information about the generation or production process of the content. This information is related to the content but it is generally not explicitly depicted by the content itself. It describes features such as the title, the author of the script, the director, the character, the target audience, the genre, reviews, rating and so forth. The usage information describes rights, financial aspects and availability of the content.

In the simplest case, the content will be created or recorded only once. In this situation, the content can be represented as a unique media instance with its associated format, coding scheme, creation information and usage information. A typical example of this simplest case is a picture from a consumer digital photograph album (see Figure 8.1). Nevertheless, more complex scenarios have to be considered, for example, where a single event, called a *reality* (e.g. a sports event), is recorded in different modalities or with different encoding formats. In this case, different content entities will be produced. The media identification tools describe each content entity by providing a unique identifier for the content entity and the audiovisual domain information. For example, one may record the reality of Ana and Alex shaking hands with a digital photo camera and a digital video camera (see Figure 8.2, where these two content entities are illustrated). In this case, the two content entities correspond to the same reality. The reality captured by one or more content entities can then be described as one or more narrative worlds, as defined in Section 8.4, using the semantic tools presented in that section.

Each content entity can also be encoded with various coding schemes or parameters and stored in various formats. The combination of a coding scheme with its associated parameters and a format corresponds to a media profile. The concepts of a media profile refers to different variations that can be produced from a master media instance by selecting various coding schemes, parameters or storage formats. Within the same media profile, several copies or instances of the same content can be created. The original media profile is called the *master* profile. If the master is digital, any copy of this instance is indistinguishable from the original and is considered as an instance of the master media profile. In the case of an analog master, copies can be considered instances of the same master media profile if the replication process causes an insignificant loss of quality. Otherwise, each copy generates a new media profile. Finally, the process of generating new instances from existing media instances by changing media formats, encoding schemes and/or parameters creates new media profiles.

The content management description shown in Figure 8.2 illustrates an image and a video content entities. The image content entity has two profiles; the master profile is of high quality and has only one instance. For applications where this quality is not necessary (e.g. for publication on a Web page), the second profile has a reduced spatial resolution of the image and a coarser quantization. This example includes two instances of the second profile. The video content entity in Figure 8.2 has only one profile with one instance but captures the same reality as the image content entity, that is, 'Ana and Alex are shaking hands in front of Columbia University, on September 19th, 1998'. These two examples also illustrate the relationships between the media, creation and usage descriptions.

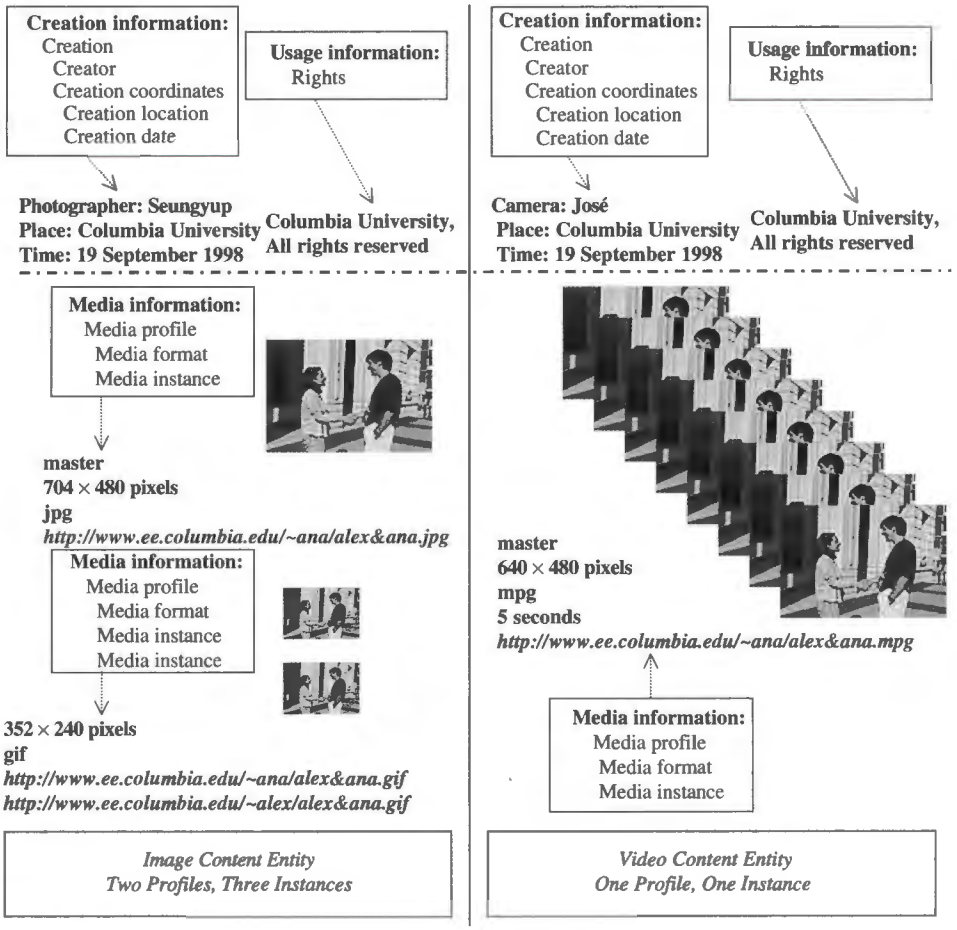


Figure 8.2 Examples of media, creation and usage descriptions of two different content entities capturing the same the reality – 'Ana and Alex are shaking hands in front of Columbia University, on September 19th, 1998'

Beside the description of media profiles and instances, content management tools can also provide transcoding hints that allow the creation of additional media variations for applications that need to adapt the content for transmission or delivery under specific network conditions or to terminals with specific characteristics. Note that the transcoding hints provide information on how to generate new content from the current piece of content. The variation tools [3] have a different purpose as they allow the description of relationships between preexisting variations of the content.

8.2.1 Media Information

The description of the media features of a content entity is wrapped in the MediaInformation Description Scheme. The MediaInformation description scheme is composed

of an optional MediaIdentification Descriptor and one or more MediaProfile description schemes.

The MediaIdentification Descriptor identifies the content entity independently of the available media profiles and associated media instances. It includes a unique identifier and optional information about the AV domain (i.e. the type of the content in terms of generation such as synthetic or natural creation, in terms of application).

The MediaProfile description scheme allows the description of one media profile of the content entity. The MediaProfile description scheme is composed of a MediaFormat Descriptor, a MediaInstance description scheme, a MediaTranscodingHints Descriptor and a MediaQuality Descriptor.

- *The MediaFormat Descriptor* describes the coding parameters and the format of a media profile. For example, content type (image, video, audio etc.), file format, file size, coding scheme, parameters describing frames, pixels, or audio samples are defined, among others.
- *The MediaInstance description scheme* identifies and locates the different instances available in the same media profile. It contains a unique instance identifier and the location of the instance. It relies on a MediaLocator or a LocationDescription element.
- *The MediaTranscodingHints Descriptor* specifies transcoding hints of the media profile being described. The purpose of this description tool is to improve the quality and reduce the complexity for transcoding applications.
- *The MediaQuality Descriptor* contains rating information about the signal quality of a media profile. The quality of audio or visual content may decrease when the signal goes through compression, transmission or signal conversion. It can describe both subjective and objective quality ratings.

Note that the use of these tools is often optional. In addition, the MediaProfile description scheme is recursive, in other words, it can contain other MediaProfile description schemes for describing the multiplexing of elementary multimedia streams to form composed content entities. Examples include MPEG-4 streams that include multiple media objects, and DVDs where multiple audio streams are available for different languages.

8.2.2 Content Creation

The creation information tools describe author-generated information about the generation or production process of a content entity. This information cannot usually be extracted from the content entity itself. That is, the information is related to, but not explicitly depicted by, the actual content entity.

The description of the creation information is wrapped in the CreationInformation description scheme, which is composed of one Creation description scheme, an optional Classification description scheme and zero or more RelatedMaterial description schemes.

- *The Creation description scheme* describes the creation of the content entity, including places, dates, actions, materials, staff (technical and artistic, for example, directors and

actors) and organizations involved. Titles and abstracts can also be described using the Creation description scheme.

- *The Classification description scheme* classifies the content for searching and filtering. It may be used in conjunction with the user-preference tools [4]. It includes user-oriented classifications such as language, subject, genre, production type (e.g. documentary and news), as well as service-oriented classifications such as purpose, parental guidance, market segment and media review.
- *The RelatedMaterial description scheme* specifies additional information about the content entity available in other materials.

8.2.3 Content Usage

The description of the usage information is wrapped in the UsageInformation description scheme, which contains an optional Rights data type, an optional Financial data type and zero or more Availability and UsageRecord description schemes.

- *The Rights data type* gives access to information about the right holders and the access rights. No rights information is explicitly described by MPEG-7. The MPEG-7 standard simply provides links to the right holders and other information related to rights management and protection. The Rights data type provides these references in the form of unique identifiers that are under management by external authorities. The underlying strategy is to enable MPEG-7 descriptions to provide access to current rights owner information without dealing with the information and negotiation directly.
- *The Financial data type* contains information related to the costs generated and incomes produced by the multimedia content. The notions of partial costs and incomes allow the classification of different costs and incomes as a function of their type. Total and subtotal costs and incomes can be calculated by the application from these partial values.
- *The Availability description scheme* describes the access availability of the content entity's media instances. It contains tools for referencing the associated media instance and describing, among others, the type of publication medium, the disseminator, additional financial information such as publication costs, price of use, and the availability period and type (e.g. live, repeat, first showing, conditional access, pay per view).
- *The UsageRecord description scheme* describes the past use of the content. It contains a reference to the associated instances of the Availability description scheme.

Note that the UsageInformation description scheme may be updated or extended each time the content is used (e.g. instances of the UsageRecord description scheme and the Income element in the Financial data type) or when there are new ways to access to the content (e.g. instances of the Availability description scheme).

8.3 CONTENT STRUCTURE

The content structure tools describe the spatial, temporal and media source structure of multimedia content, as a set of interrelated segments. They describe the segments, their attributes and the hierarchical decompositions and structural relationships among

segments. In this section, we describe in more detail the MPEG-7 segment entity, segment attribute, segment decomposition and structural relation tools.

8.3.1 Segment Entities

The MPEG-7 segment entity tools describe segments of multimedia content in space, time and/or media source. The segment entity tools are derived from the Segment description scheme, which is an abstract type that represents an arbitrary fragment or section of multimedia content. The description schemes derived from the Segment description scheme describe generic or specific types of segments.

The generic segment entity tools describe spatiotemporal segments of generic types of multimedia content such as images, videos, audio sequences and audiovisual sequences. The most basic visual segment entity tool is the StillRegion description scheme, which describes a spatial region of a 2-D image or a video frame (a still region is a group of pixels in the digital case). Figure 8.1 shows three examples of still regions in an image corresponding to the full image (whose identity is 'SR1' in Figure 8.1) and the two people depicted by the image ('SR2' and 'SR3'). Other purely visual segment entity tools are the VideoSegment description scheme and the MovingRegion description scheme, which describe, respectively, a temporal interval and a spatiotemporal region of a video sequence (a group of frames in a video and a group of pixels in a group of video frames, respectively, in the digital case). The AudioSegment description scheme represents a temporal interval of an audio sequence (a group of samples in the digital case). Both the AudioVisualSegment description scheme and the AudioVisualRegion description scheme describe audio and visual data in an audiovisual sequence. In particular, the AudioVisualSegment description scheme describes a temporal interval of audiovisual data, which corresponds to both the audio and video in the same temporal interval, whereas the AudioVisualRegion description scheme describes an arbitrary spatiotemporal segment of audiovisual data, which corresponds to the audio in an arbitrary temporal interval and the video in an arbitrary spatiotemporal region. Figure 8.3 shows examples of a video segment, a moving region, an audiovisual segment and several audiovisual regions.

There is also a set of MPEG-7-specific segment entity tools that describe segments of specific multimedia content types such as mosaic, 3-D, image- or videotext, ink content, multimedia content and video editing. For example, the Mosaic description scheme extends from the StillRegion description scheme and describes a mosaic or panoramic view of a video segment. A mosaic is usually constructed by aligning and blending together the frames of a video segment upon each other using a common spatial reference system. The StillRegion3D description scheme represents a 3-D spatial region of a 3-D image. The ImageText description scheme and the VideoText description scheme extend, respectively, from the StillRegion description scheme and the MovingRegion description scheme. They describe a still region and a moving region that correspond to the text, respectively. The InkSegment description scheme represents a spatiotemporal segment of ink content created by a pen-based system or an electronic white board. The MultimediaSegment description scheme represents a composite of segments forming a multimedia presentation such as an MPEG-4 presentation or a Web page. Finally, the analytic edited video segment description schemes such as the AnalyticClip description scheme and the

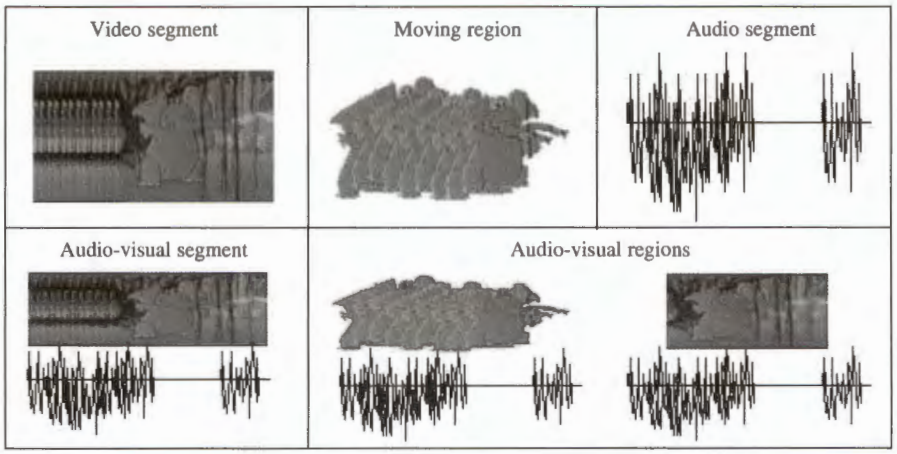


Figure 8.3 Examples of a video segment, a moving region, an audio segment, an audiovisual segment, an audiovisual region composed of a moving region and an audio segment and an audiovisual region composed of a still region

AnalyticTransition description scheme extend originally from the VideoSegment description scheme and describe, respectively, different types of shots and transitions between the shots resulting from video editing work. The video editing description is 'analytic' in the sense that it is made a posteriori on the final edited video content. Figure 8.4 shows examples of a mosaic, a 3-D still region, an image text, an ink segment, a multimedia segment, two analytic clips and an analytic transition.

The segment entity tools can describe segments that are not connected but composed of several separated connected components. Connectivity refers here to both spatial and temporal dimensions. A temporal segment (instance of the VideoSegment, AudioSegment, AudioVisualSegment, InkSegment description schemes and the audio part of the

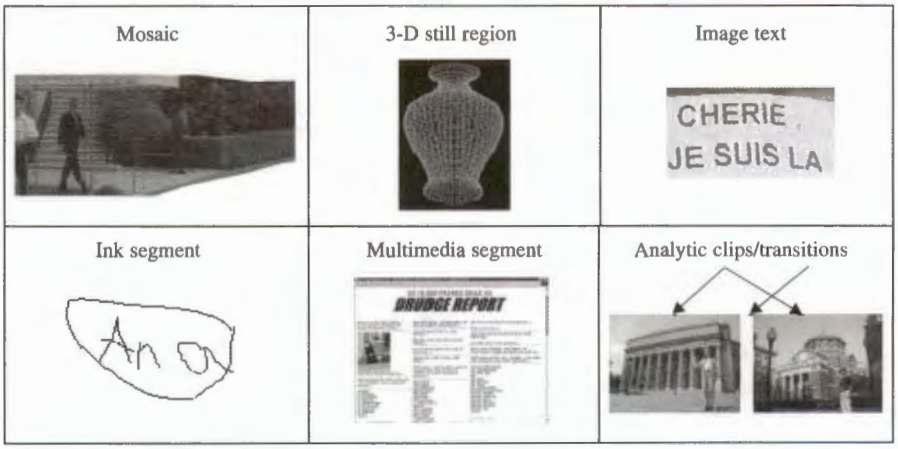


Figure 8.4 Examples of a mosaic, a 3-D still region, an image text, an ink segment, a multimedia segment, two analytic clips and an analytic transition

AudioVisualRegion description scheme) is said to be temporally connected if it is continuous over a temporal interval (e.g. a sequence of continuous video frames and/or audio samples). A spatial segment (instance of the StillRegion description scheme) is said to be spatially connected if it is a continuous spatial region (e.g. a group of connected pixels). A spatiotemporal segment (instance of the MovingRegion description scheme and the video part of the AudioVisualRegion description scheme) is said to be spatially and temporally connected if it appears in a temporally connected segment and if it is formed of a spatially connected segment at each time instant. Note that this is not the classical definition of connectivity in a 3-D space.

Figure 8.5 illustrates several examples of temporal and spatial segments that are connected or composed of several connected components. Figure 8.6 shows examples of a connected and a nonconnected moving region. In Figure 8.6(b), the moving region is not connected because it is not instantiated in all the frames and, furthermore, it involves several spatial connected components in some of the frames.

The Segment description scheme is abstract and therefore cannot be instantiated on its own. However, the Segment description scheme contains elements and attributes that are common to the different segment types. Among the common properties of segments, there is information related to the creation, usage, media, text annotation and importance for matching and point of view. Derived segment entity tools can describe other visual, audio and spatiotemporal properties of segments. Note that in all cases, the description tools attached to the segment entity tools are global to the union of the connected components composing the segment being described. At this level, it is not possible to describe the individual connected components of the segment. If connected components have to be described individually then the segment has to be decomposed into various subsegments

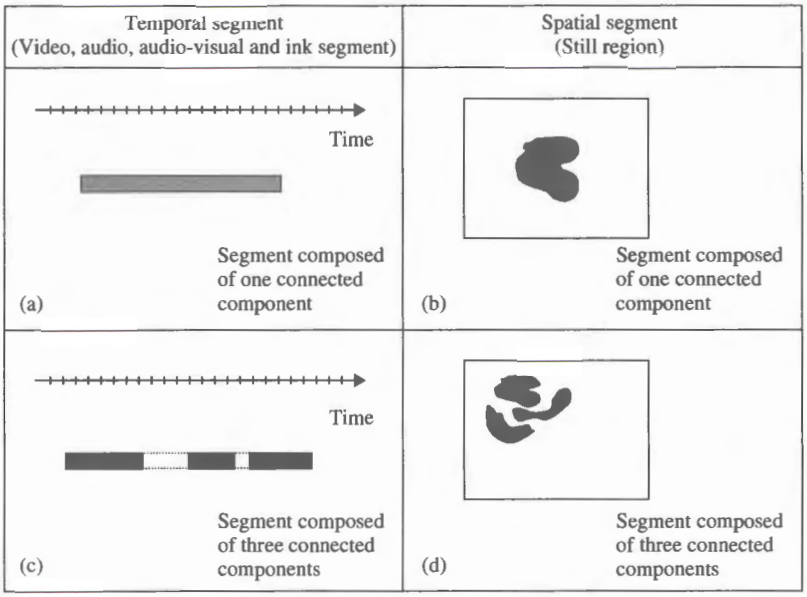


Figure 8.5 Examples of segments: (a) and (b) segments composed of one single connected component; (c) and (d) segments composed of three separated connected components

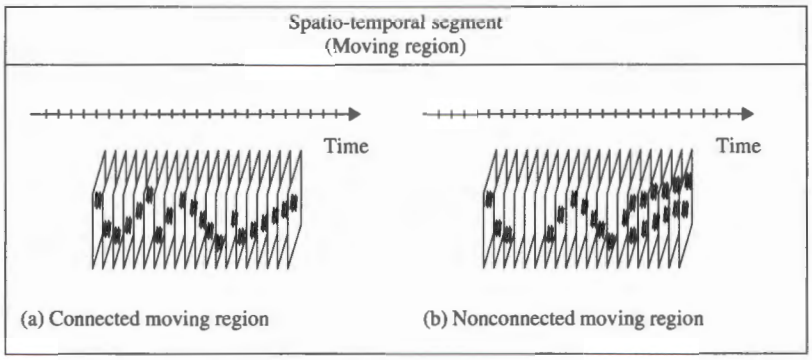


Figure 8.6 Examples of connected: (a) nonconnected and (b) moving region

corresponding to its individual connected components using the segment decomposition tools (see Section 8.3.3).

8.3.2 Segment Attributes

As mentioned before, any kind of segment can be described in terms of its media information, creation information, usage information, text annotations, visual features, audio features and other segment attributes. Specific features such as the characterization of the connected components of the segment, the segment importance, the importance of some of its descriptors can also be described. As an example, the properties of the still regions in the image description in Figure 8.1 are described using text annotation, color structure and matching hint tools.

The Mask Descriptor is an abstract type that represents the connected components of a segment in time, space and media, among others. The SpatialMask, TemporalMask and SpatioTemporalMask Descriptors extend the Mask Descriptor to describe the localization of separated connected components (subregions or subintervals) in space and/or in time. They are used to describe the connectivity of nonconnected segments (see Section 8.3.1). For example, the SpatialMask Descriptor describes the localization of the spatial connected components of a still region. Similarly, the TemporalMask Descriptor describes the localization of the temporal connected components of a video segment, an ink segment, an audio segment, an audiovisual segment, or the audio part of an audiovisual region. Finally, the SpatioTemporalMask Descriptor describes the localization of the spatiotemporal components of a moving region or the visual part of an audiovisual region. In the InkSegment description scheme, the SceneGraph, MediaSpaceMask and OrderedGroupDataSetMask Descriptors, which also extend the Mask Descriptor, describe the localization of separated subgraphs in a scene graph, data subintervals in a file and subsets in an ordered group data set for ink segments.

The importance of segments and segment descriptors is described by the MatchingHint and PointOfView Descriptors, respectively. The MatchingHint Descriptor describes the relative importance of instances of audio or visual descriptors or description schemes (e.g. the color structure of a still region) or parts of audio or visual descriptors or description schemes (e.g. the fourth Value element of a color structure description of a still

region) in instances of segment entity tools. For specific applications, the MatchingHint Descriptor can improve the retrieval performance because the most relevant descriptors for matching may depend on the application but also may vary from segment to segment. On the other hand, the PointOfView Descriptor describes the relative importance of segments given a specific viewpoint. The PointOfView Descriptor assigns values between zero and one to segments on the basis of a viewpoint specified by a string (e.g. 'Home team' for a soccer game).

Other segment attribute tools describe specific media, creation and handwriting recognition information related to ink segments. These tools are the InkMediaInfo, HandWritingRecogInformation and HandWritingRecogResult description schemes. The InkMediaInformation description scheme describes parameters of the input device of the ink content (e.g. width, height and lines of the writing field's bounding box), writer's handedness (e.g. 'right' or 'left') and ink data style (e.g. 'cursive' or 'drawing'). The HandWritingRecogInformation description scheme describes the handwriting recognizer and the ink lexicon used by the recognizer. The HandWritingRecogResult description scheme describes handwriting recognition results, in particular, the overall quality and some results of the recognition process with corresponding accuracy scores.

Finally, depending on the nature of the segment, visual and audio descriptors or description schemes can be used to describe specific features related to the segment. Examples of visual features are color, shape, texture and motion, whereas examples of audio features involve audio spectrum, audio power, fundamental frequency, harmonicity, timbre, melody and spoken content, among others.

8.3.3 Segment Decompositions

The segment decomposition tools describe the decomposition or subdivision of segments into subsegments. They support the creation of segment hierarchies to generate, for example, tables of contents. The decomposition is described by the SegmentDecomposition description scheme, which is an abstract type that represents an arbitrary decomposition of a segment. Segment decomposition tools derived from the SegmentDecomposition description scheme describe basic types of segment decompositions and the decompositions of specific types of segments. It is important to note that MPEG-7 does not specify how a segment should be divided into subsegments or describe the segmentation process; however, it can describe the criteria used during the segmentation and the dimensions affected by the segmentation: space, time and/or media source.

The basic segment decomposition tools are abstract types that describe spatial, temporal, spatiotemporal and media source decompositions of segments. These tools are the SpatialSegmentDecomposition, TemporalSegmentDecomposition, SpatioTemporal SegmentDecomposition and MediaSourceDecomposition description schemes, accordingly. For example, an image can be decomposed spatially into a set of still regions corresponding to the objects in the image, which, at the same time, can be decomposed into other still regions. Figure 8.1 exemplifies the spatial decomposition of a still region into two still regions. Similar decompositions can be generated in time and/or space for video and other multimedia content. Media source decompositions divide segments into its media constituents such as audio tracks or viewpoints from several cameras.

The subsegments resulting from decomposition may overlap in time, space and/or media source; furthermore, their union may not cover the full time, space and media extents of the parent segment, thus leaving gaps. Two attributes in the SegmentDecomposition description scheme indicate whether a decomposition leaves gaps or overlaps. Note that, in any case, the segment decomposition implies that the union of the spatiotemporal and media spaces defined by the child segments is included in the spatiotemporal and media spaces defined by their parent segment (i.e. children are contained in their parents).

Several examples of decompositions for temporal segments are included in Figure 8.7. Figure 8.7(a) and (b) show two examples of segment decompositions with neither gaps nor overlaps (i.e. a partition in the mathematical sense). In both cases, the union of the temporal extents of the children corresponds exactly to the temporal extent of the parent, even if the parent is itself nonconnected (see Figure 8.7b). Figure 8.7(c) shows an example of decomposition with gaps but no overlaps. Finally, Figure 8.7(d) illustrates a more complex case in which the parent is composed of two connected components and its decomposition generates three children with gaps and overlaps: the first child is itself nonconnected and composed of two connected components, the two remaining children are composed of a single connected component.

The decomposition of a segment may result in segments of different nature. For example, a video segment may be decomposed into other video segments as well as into still or moving regions. However, not all the combinations of parent segment type, segment decomposition type and child segment type are valid. For example, an audio segment can only be decomposed in time or media into other audio segments; the spatial decomposition of an audio segment into still regions is not valid. The valid decompositions

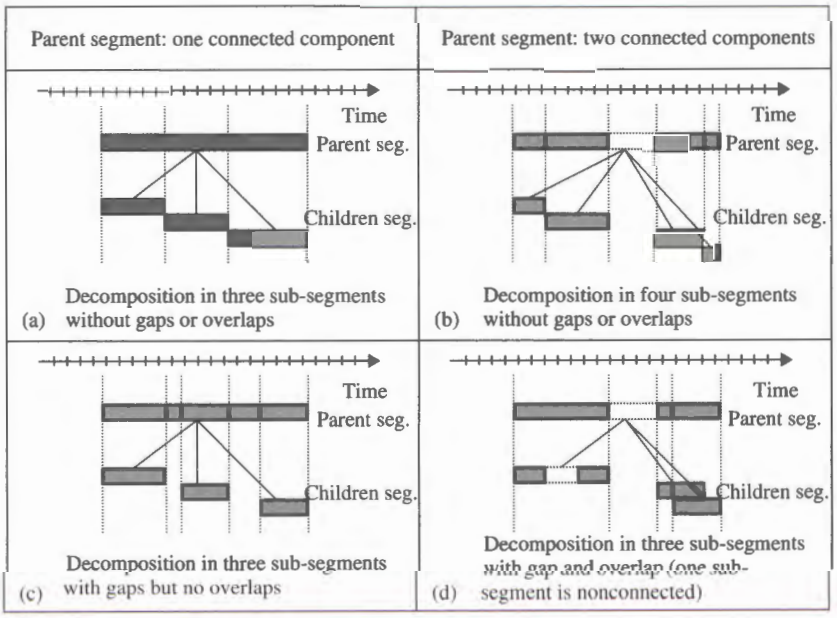


Figure 8.7 Examples of segment decompositions: (a) and (b) decompositions with neither gaps nor overlaps; (c) and (d) decompositions with gaps and/or overlaps

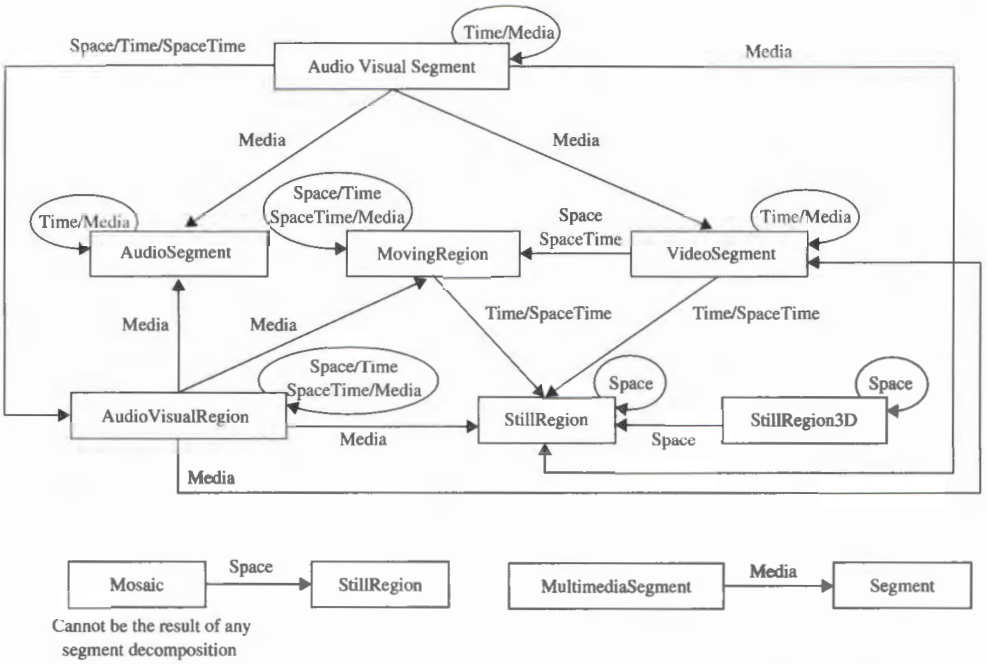


Figure 8.8 Valid segment decompositions among some specific types of segments. 'time', 'space', 'spaceTime' and 'media' stand for temporal, spatial, spatiotemporal and media source segment decomposition, respectively. Each description scheme represents itself and its derived description schemes

among some specific types of segments are shown in Figure 8.8. Each description scheme in the figure represents itself as well as its derived description schemes. For example, as shown in the figure, a still region can only be decomposed in space into other still regions including image texts. The Mosaic description scheme is a special case of the StillRegion description scheme, which cannot be the result of any segment decomposition.

An example of image description with several spatial decompositions is illustrated in Figure 8.9. The full image (whose identity is 'SR1') is described using the StillRegion description scheme whose creation (title, creator), usage (copyright), media (file format), text annotation (textual description) and color properties are described using the CreationInformation, UsageInformation, MediaInformation, TextAnnotation and ColorStructure description tools. This first still region is decomposed into two still regions, which are further decomposed into other still regions. For each decomposition step, Figure 8.9 indicates whether gaps and overlaps are generated. The complete segment hierarchy is composed of nine still regions (note that 'SR9' and 'SR7' are single segments composed of two separated connected components). For each region, Figure 8.9 shows the type of feature that is described.

Describing the hierarchical decomposition of segments is useful to design efficient search strategies (global search to local search). It also allows the description to be scalable: a segment may be described by its direct set of description tools, but it may also be described by the union of the description tools that are related to its subsegments.

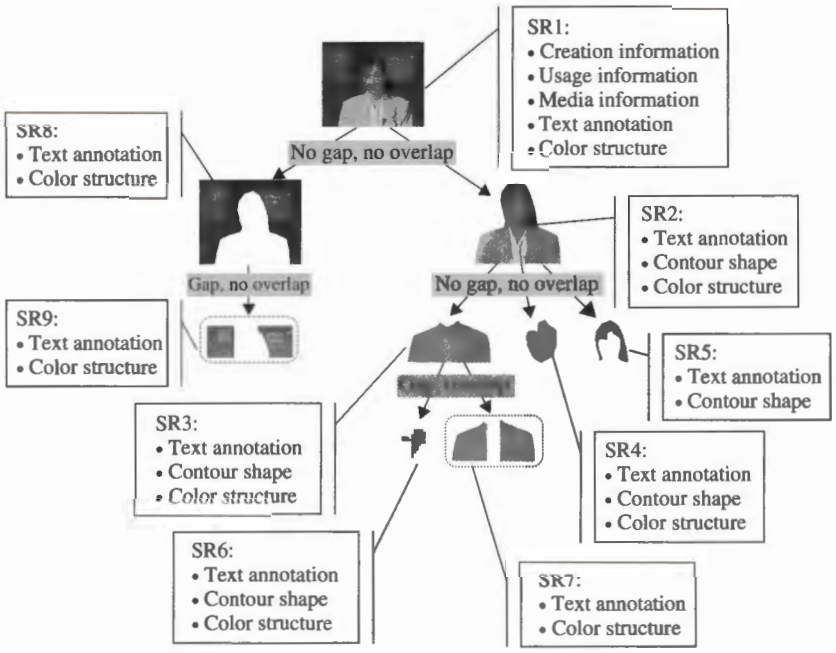


Figure 8.9 Example of an image description as a hierarchy of still regions and associated features

8.3.4 Structural Relations

The description of the content structure in MPEG-7 is not constrained to rely on hierarchies. Although, hierarchical structures provided by the segment decomposition tools are adequate for efficient access, retrieval and scalable description, they imply constraints that may make them inappropriate for certain applications. In such cases, the MPEG-7 graph, base and structural relation tools can be used to describe more general segment structures. MPEG-7 has standardized common structural relations such as *left* and *precedes* but it allows the description of non-normative relations (see soccer example below). Figure 8.1 shows an example of the spatial relationship *left* between the still regions 'SR2' and 'SR3'.

The structural relation tools include the SpatialRelation Classification Scheme (CS) and the TemporalRelation CS, which specify spatial and temporal relations, respectively. The normative structural relations in MPEG-7 are listed in Table 8.1. For each normative structural relation, MPEG-7 has also standardized the inverse relation. Note that the

Table 8.1 Normative structural relations in MPEG-7 listed by type

Type	Normative relations
Spatial	South, north, west, east, northwest, northeast, southwest, southeast, left, right, below, above, over, under
Temporal	Precedes, follows, meets, metBy, overlaps, overlappedBy, contains, during, strictContains, strictDuring, starts, startedBy, finishes, finishedBy, coOccurs, contiguous, sequential, coBegin, coEnd, parallel, overlapping

inverse relation is defined as follows: 'A InverseRelation B' ⇔ 'B Relation A', where A and B are the source and the target of *InverseRelation*, respectively.

The SpatialRelation CS specifies spatial relations that apply to entities that have 2D spatial attributes, which may be implicitly or explicitly stated. Normative spatial relations describe how entities are placed and relate to each other in 2-D space (e.g. *left* and *above*). The normative spatial relations may apply to entities such as still regions (see Section 8.3.1) and objects (see Section 3.2), among others.

In a similar way, the TemporalRelation CS specifies temporal relations that apply to entities that have implicitly or explicitly stated temporal attributes. Normative temporal relations include binary relations between two entities corresponding to Allen's temporal interval relations [5] (e.g. *precedes* and *meets*) and n-ary relations among two or more entities such as *sequential* and *parallel*, among others. The normative temporal relations may apply entities such as video segments (see Section 8.3.1), moving regions (see Section 8.3.1) and events (see Section 8.4.2), among others.

Finally, the generic relations specified by the GraphRelation CS (e.g. *identity* [1]) and the BaseRelation CS (e.g. *union* and *disjoint* [1]) can also be used to describe relations among segments.

To illustrate the use of the segment entity and structural relation tools, consider the example in Figure 8.10 and Figure 8.11. This example shows an excerpt from a soccer match and its description. Two video segments and four moving regions are described. A graph of structural relationships describing the structure of the content is shown in Figure 8.11. The video segment 'Dribble&Kick' involves the moving regions 'Ball', 'Goalkeeper' and 'Player'. The moving region 'Ball' remains *close to* the moving

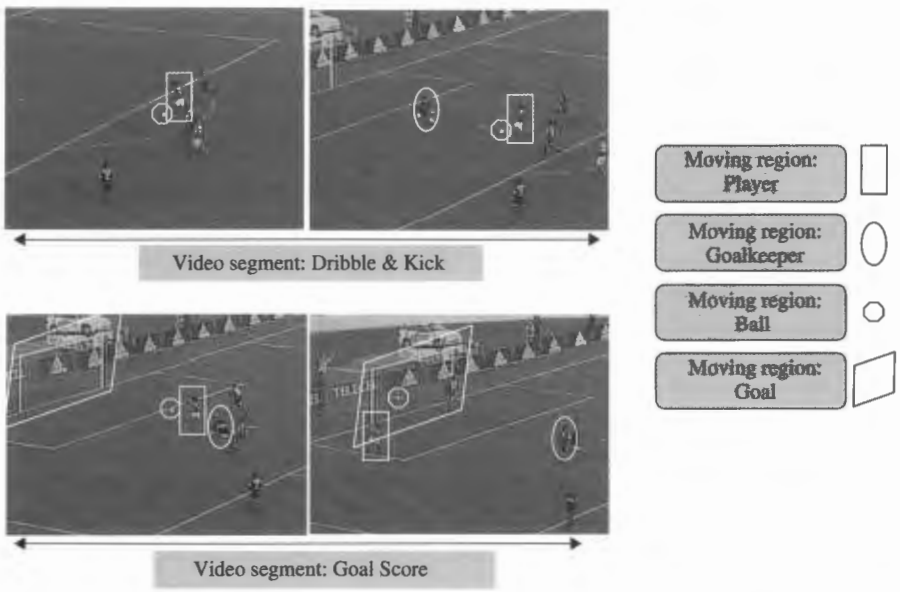


Figure 8.10 Excerpt from a soccer match with the corresponding video segments and moving regions that participate in the graph in Figure 8.11

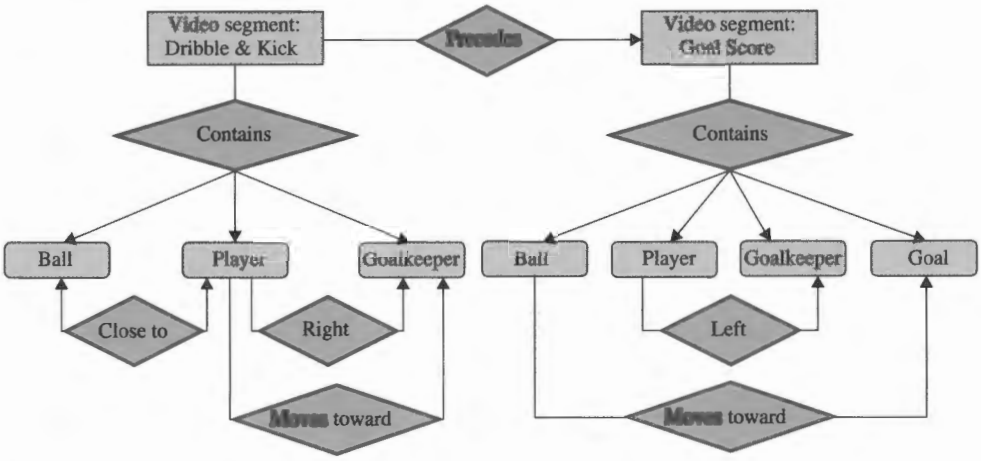


Figure 8.11 Example of a graph of structural relationship for the video segments and moving regions in Figure 8.10

region 'Player', which is *moving toward* the moving region 'Goalkeeper'. The moving region 'Player' is on the *right* of the moving region 'Goalkeeper'. The video segment 'GoalScore' involves three moving regions corresponding to the same three objects as the three previous moving regions including the moving region 'Goal'. In this video segment, the moving region 'Player' is on the *left* of the moving region 'Goalkeeper' and the moving region 'Ball' *moves toward* the moving region 'Goal'. This example illustrates the flexibility of this kind of representation. Note that this description is mainly structural because the relationships specified in the graph edges are purely spatiotemporal and visual and the nodes represent segments. The only explicit semantic information in the description is available from the text annotation (where key words such as 'Ball', 'Player', or 'Goalkeeper' are used). All the relations in this example are normative except for *close to* and *moves toward.*

8.4 CONTENT SEMANTICS

The MPEG-7 tools for describing the semantics of multimedia content can be best understood by analyzing how semantic descriptions of anything are constructed in general. One way to describe semantics is to start with events, understood as occasions when something happens. Objects, people and places can populate such occasions and the times at which they occur. Some descriptions may concentrate on the latter aspects, with or without events, and some may even begin with a place, or an object, and describe numerous related events. Furthermore, these entities can have properties and states through which they pass as what is being described transpires. There are the interrelations among these entities. Finally, there is the world in which all of this is going on, the background, the other events and other entities, which provide context for the description.

If a description is being given between people, much of the information exchange is done with reference to common experiences, past events and analogies to other situations. A semantic description may be more or less complete, and may rely on a little or a lot

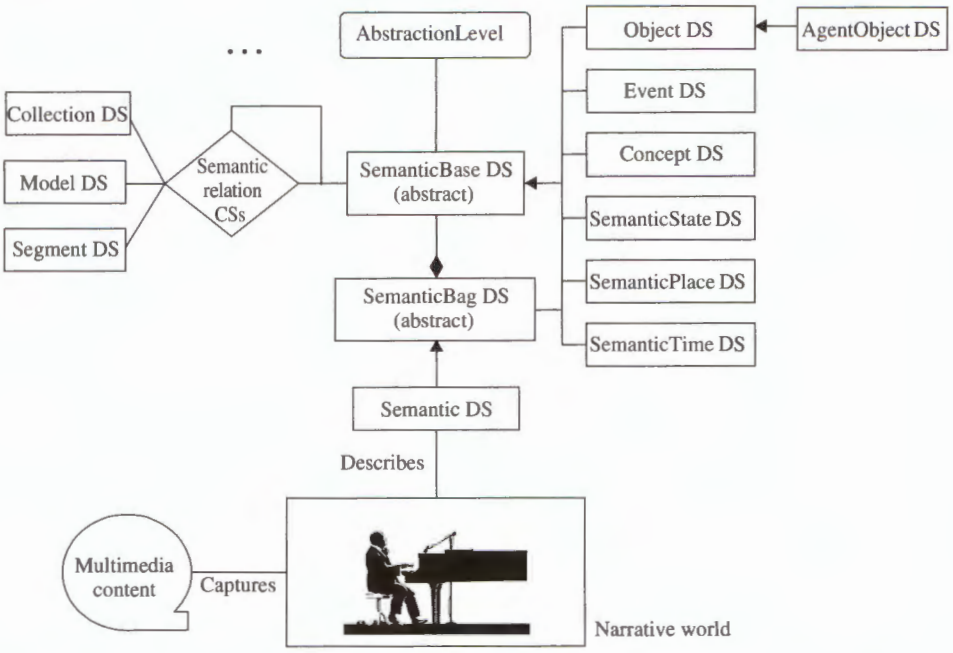


Figure 8.12 MPEG-7 tools that describe the semantics of multimedia content. The figure makes use of Unified Modeling Language (UML) and Extended Entity Relation (EER) graphical formalisms. Rectangles, rounded-edge rectangles and diamonds represent entities, attributes and relations, respectively; the edges that start with shaded triangles and shaded diamonds represent inheritance and composition, respectively; lines with names represent associations

of generic or contextual information and on other descriptions. Since the ultimate goal is to describe the semantics of multimedia content and since much of these semantics are narrative in quality (as in a movie that tells a story or an Expressionist symphony), MPEG-7 refers to the participants, background, context and all the other information that makes up a single narrative as a 'narrative world'. One piece of content may have multiple narrative worlds or vice versa, but the narrative worlds exist as separate semantic descriptions.

The components of the above semantic descriptions roughly fall into entities that populate narrative worlds, their attributes and the relationships among them. In this section, we describe in more detail the tools provided by MPEG-7 to describe such semantic entities, attributes and relationships, which are shown in Figure 8.12. We start by introducing the abstraction model of MPEG-7 semantic descriptions and finish by discussing application issues in using the MPEG-7 content semantic tools for implicitly or explicitly describing the semantics of multimedia content.

8.4.1 Abstraction Model

Besides the concrete semantic description of specific instances of multimedia content, MPEG-7 content semantic tools also allow the description of abstractions and abstract quantities.

Abstraction refers to taking a semantic description of a specific instance of multimedia content (e.g. 'Alex is shaking hands with Ana' in the image shown in Figure 8.1) and generalizing it either to multiple instances of multimedia content (media abstraction, e.g. 'Alex is shaking hands with Ana' for any picture or video including the images and videos shown in Figure 8.2) or to a set of concrete semantic descriptions (formal abstraction, e.g. 'Alex is shaking hands with any woman' or 'Any man is shaking hands with any woman'). When dealing with the semantics of multimedia content, it becomes necessary to draw upon descriptive techniques such as abstraction because these techniques are part of the way the world is understood and described by humans. MPEG-7 considers two types of abstraction, which are called *media abstraction* and *formal abstraction*. Moreover, MPEG-7 also provides ways of representing abstract properties and concepts that do not arise by a media or a formal abstraction. As examples, Figure 8.1 shows a concrete description of an image that contains an abstract property or concept 'comradeship' and Figure 8.13 illustrates a media abstraction and a formal abstraction of the description in Figure 8.1.

An abstraction, or more precisely, a lambda abstraction, in logic, creates a function from a statement [6]. This process is effected by replacing one or more of the constant expressions in the statement by a variable. There are two types of models for such abstraction, typed and untyped. The more useful method is typed abstraction. For this abstraction, the variable is given a type, such as a data type, and the constants that may be substituted for it must be of this type. In this process, the statement is changed into

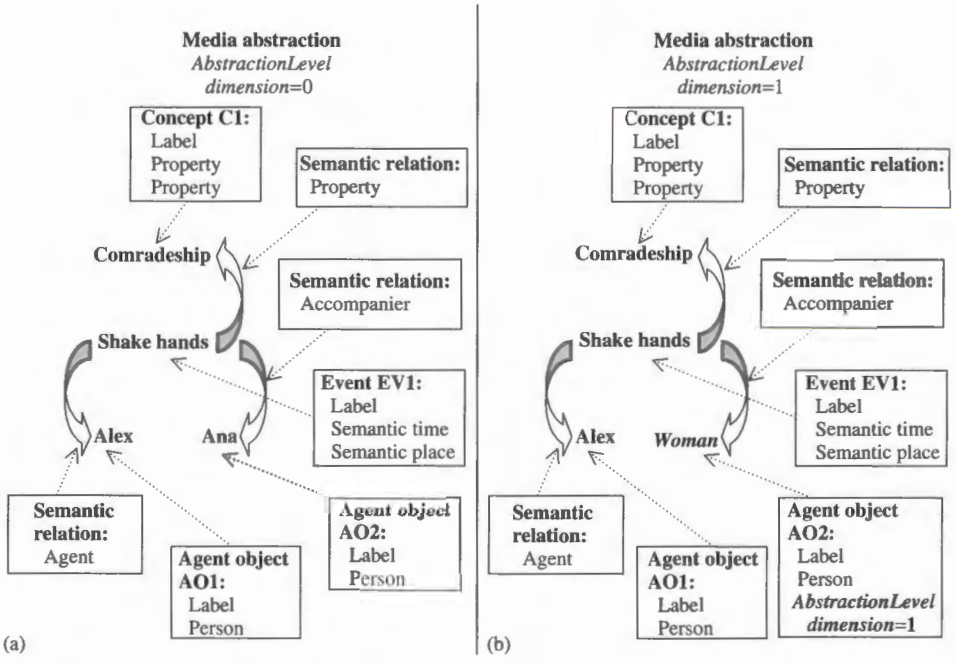


Figure 8.13 Examples of a media abstraction: (a) formal abstraction and (b) for the semantic description in Figure 8.1. Section 8.4.3 explains how to use the AbstractionLevel data type to indicate media and formal abstractions

a set of statements, one for each choice of constants. Abstraction thus replaces a single instance with a more general class. In describing media, two types of abstractions can occur: media abstractions and formal abstractions. The method for indicating these two types of abstractions is to use the dimension attribute in the AbstractionLevel data type (see Section 8.4.3).

A media abstraction is a description that has been separated from a specific instance of multimedia content and can describe all instances of multimedia content that are sufficiently similar (similarity depends on the application and on the detail of the description). If we consider the description 'Alex is shaking hands with Ana' of the image shown in Figure 8.1, a media abstraction is the description 'Alex is shaking hands with Ana' with no links to the image and that is, therefore, applicable to any image or video depicting that event (see Figure 8.13a). Another example is a news event broadcast on different channels to which the same semantic description applies. The variable in a media abstraction is the media itself.

A formal abstraction describes a pattern that is common to a set of examples. The common pattern contains placeholders or variables to be filled in that are common to the set. A formal abstraction may be formed by gathering a set of examples, determining the necessary placeholders from these examples and thus deriving the pattern, or by first creating the common pattern, and then creating the specific examples from the placeholder. The description is a formal abstraction as long it contains variables, which, when filled in, would create specific examples or instances, either media abstractions or concrete descriptions. Formal abstractions of the semantic description in Figure 8.1 could be 'Alex is shaking hands with any woman' and 'Any man is shaking hands with any woman', where 'any woman' and 'any man' are the variables of the formal abstractions (see Figure 8.13b). The variables in a formal abstraction can be replaced or filled in with concrete instances using the semantic relation *exemplifies* (see Section 8.4.4).

MPEG-7 also provides methods of describing abstract quantities such as properties (see the Property element in Section 8.4.3) and concepts (see the Concept description scheme Section 8.4.2). For instance, the property 'ripeness' can be an attribute of the object 'banana'. Concepts are collections of properties that define a category of entities but cannot fully characterize it, that is, abstract quantities that are not the result of an abstraction 'comradeship' (see semantic description in Figure 8.1). An abstract quantity may be describable only through its properties, and therefore is not abstract because it comes from the replacement of elements of a description by generalizations or variables. Note that the same abstract quantity could appear as a property or as a concept in a description. In order to describe relationships of one or more properties, or allow multiple semantic entities to be related to one or more properties, or specify the strength of one or more properties, the property or collection of properties must be described as a concept using the Concept description scheme.

8.4.2 Semantic Entities

The MPEG-7 semantic entity tools describe semantic entities such as narrative worlds, objects, events, concepts, states, places and times. The semantic entity tools are derived from the SemanticBase description scheme, which is an abstract type that represents

any semantic entity. The description schemes derived from the SemanticBase description scheme describe specific types of semantic entities (see Figure 8.12) such as narrative worlds, objects and events.

A narrative world in MPEG-7 is represented using the Semantic description scheme, which is described by a number of semantic entities and the graphs of their relationships. The Semantic description scheme is derived from the SemanticBag description scheme, which is an abstract type representing any kind of collection of semantic entities and their relationships. The Semantic description scheme represents one type of those collections, especially describing a single narrative world (e.g. world 'Earth' in a world history documentary or world 'Roman Mythology' in the engravings on a Roman vessel).

Specific types of semantic entities are described by description schemes derived from the SemanticBase description scheme. This abstract type holds common functionalities shared in describing any semantic entity: labels used for categorical searches, a textual description, properties, links to and descriptions of the media and relationships to other entities. In order to facilitate the use of a description of one narrative world as context for another or to allow a narrative world to be embedded in another, the Semantic description scheme (via the SemanticBag description scheme) is also derived from the SemanticBase description scheme. Other description schemes derived from the SemanticBase description scheme are the Object, AgentObject, Event, SemanticPlace, SemanticTime, SemanticState and Concept description schemes. These represent entities that populate the narrative world such as an object, agent and event; the where and when of things; a parametric entity; and an abstract collection of properties, respectively.

The Object and Event description schemes describe perceivable semantic entities, objects and events that can exist or take place in time and space in narrative worlds. The Object description scheme and Event description schemes are recursive to describe the subdivision of objects and events into other objects and events without identifying the type of subdivision. The AgentObject description scheme extends from the Object description scheme to describe an object that acts as a person, a group of persons, or an organization in a narrative world. For example, the semantic description in Figure 8.1 shows one event and two agent objects corresponding, respectively, to 'Shake hands' (whose identity is 'EV1' in Figure 8.1) and to the two people 'Alex' and 'Ana' ('AO1' and 'AO2', respectively) depicted by the image.

The SemanticPlace and SemanticTime description schemes describe a location and time in a narrative world, respectively. The event 'Shake hands' in Figure 8.1 has associated a location and a time. The SemanticPlace and SemanticTime description schemes can also describe lengths and durations. Notice that these are semantic descriptions of place or time, not necessarily numerical in any sense. They serve to give extra semantic information beyond pinpointing a location or instant. The SemanticPlace and SemanticTime description schemes can describe locations and times that are fictional or that do not correspond to the locations and times where the multimedia content was recorded, which are described using the CreationInformation description scheme. For example, a movie could have been recorded in Vancouver during winter (CreationInformation description scheme) but the action is supposed to take place in New York during fall (Semantic description scheme).

The SemanticState description scheme describes the state or parametric attributes of a semantic entity at a given place and/or time in the narrative world. The SemanticState description scheme can describe the changing of an entity's attributes' values in space and time. An example of the state of an object is the piano's weight; an example of the state of a time is the cloudiness of a day, which varies with time during the day. In a description that is being updated (e.g. a running description of a news event), the SemanticState description scheme can be used to give the current state of a semantic entity. It can also be used to parametrize the strength of a relationship between entities (e.g. the ripeness of a banana).

Finally, the Concept description scheme forms one part of the semantic description abstraction model. Using the Concept description scheme, concepts are described as collections of one or more properties and given the status of a semantic entity in their own right. As described in Section 8.4.1, the model allows abstraction of the media and abstraction of semantic entities. These are formal abstractions in the sense that they take specific elements of a description and replace them with variables or placeholders. Two other elements of abstraction involve properties of semantic entities. Properties can be modifiers to existing elements of the description. But one can also create collections of properties. In some cases, this leads to an abstraction, that is, there is a category of entities for which this collection is a definition. It is also possible that such a collection of properties does not define a category, but rather is distributed across many categories, not confined to any one. This collection is then an abstract entity that did not occur as the result of some abstraction. We call such an abstraction a concept. An example of a concept could be any affective property of the media, for instance, 'suspense', or a quantity not directly portrayed by the media, such as 'ripeness'. A concept can also represent the embodiment of an adjectival quality. Examples of such use are 'comradeship', 'happiness' and 'freedom'. Concept 'comradeship' (C1) participates in the semantic description shown in Figure 8.1.

Figure 8.14 illustrates most types of semantic entity tools in the description of a piano recital video. The textual description of the example in Figure 8.14 could be as follows: 'Tom Daniels, who is a musician, plays the piano harmoniously at Carnegie Hall from 7 to 8 P.M. on October 14, 1998, in memory of his tutor. The piano weights 100 Kg.'. The Object description scheme is used to describe the object 'piano'. 'Tom Daniels', 'Tom's tutor' and 'musician' are described using the AgentObject description scheme. The Event and Concept description schemes describe the event 'Play' and the concept 'harmony', respectively. Finally, 'Carnegie Hall' and '7 to 8 P.M., October 14, 1998' are described using the SemanticTime and SemanticPlace description schemes, respectively. Figure 8.14 also demonstrates some normative semantic relations that are presented in Section 8.4.4.

Semantic entities involved in descriptions of narrative worlds can change status or functionality, and therefore be described by different semantic entity tools. For instance, a picture can be an object (when it hangs in a museum), or a location ('In the picture'), or the narrative world itself (when describing what picture depicts 'In the picture Caesar is talking to Brutus'). If the same entity is described by multiple semantic entity tools, these descriptions can be related with the semantic relation *identifier* (see Section 8.4.4).

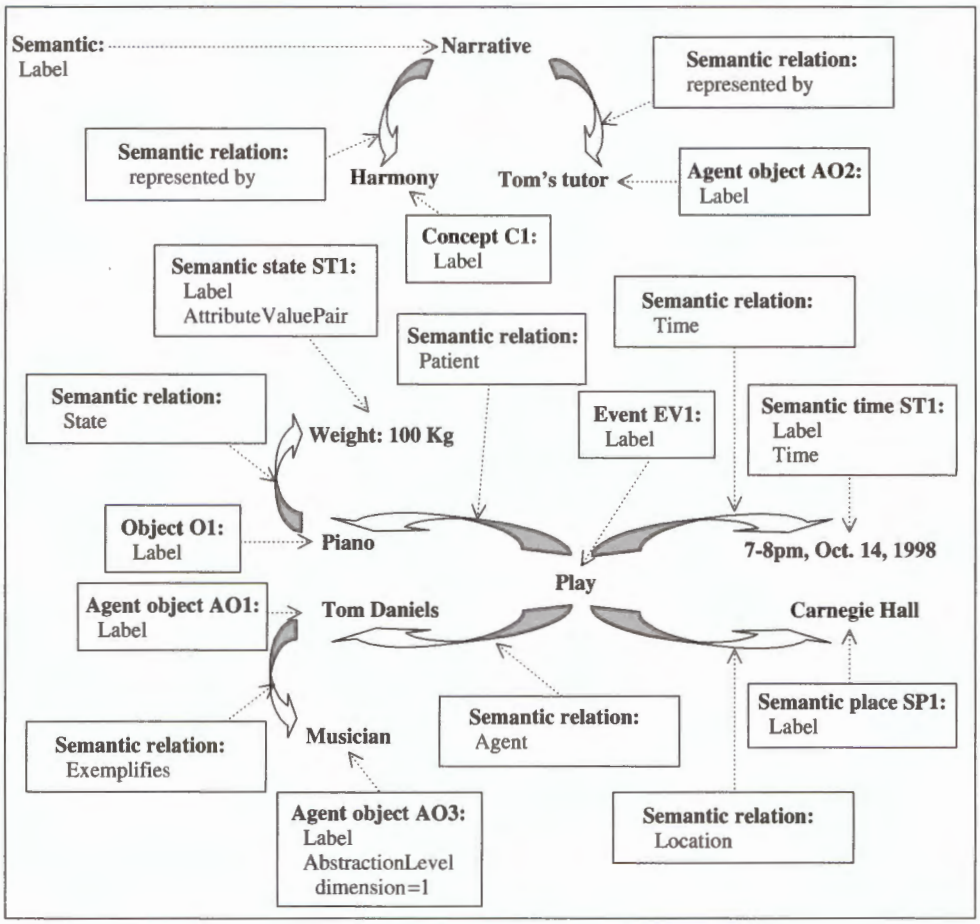


Figure 8.14 Example of a semantic description of a piano recital video

8.4.3 Semantic Attributes

In MPEG-7, semantics entities can be described by labels, by a textual definition, or in terms of properties or of features of the media or segments in which they occur. Other semantic attribute tools describe abstraction levels and semantic measurements in time and space. In addition, the Agent description scheme, the Place description scheme and the Time data type presented in [8] are used to characterize agent objects, semantic places and semantic times, respectively.

The SemanticBase description scheme contains one or more Label elements, an optional Definition element and zero or more Property elements, among others. The Label element assigns a label to a semantic entity. A label is similar to a 'descriptor' or 'index term' in Library and Information Science. It is a type used for classifying and retrieving the descriptions of semantic entities (e.g. 'man' for the agent object 'Alex' in Figure 8.1). A semantic entity can have multiple labels, one for each 'index term'. The labels can be used to retrieve all the semantic entities sharing the

same label(s). The Definition element provides a textual definition of a semantic entity. The textual definition of a semantic entity can be done with either free text or structured text (e.g. the textual description of the agent object 'Alex' in Figure 8.1 could be 'bipedal primate mammal (Homo sapiens) that is anatomically related to the great apes but distinguished especially by notable development of the brain'). The Property element describes a quality or adjectival property associated with a semantic entity (e.g. 'tall' and 'slim' for the agent object 'Alex' in Figure 8.1). This is one way of describing abstract quantities in MPEG-7, the other being the Concept description scheme, as described in Section 8.4.2. The MediaOccurrence element in the SemanticBase description scheme describes the appearance or occurrence of a semantic entity in one location in the multimedia content with a media locator, locators in space and/or time, audio descriptors or description schemes and visual descriptors or description schemes. The purpose of the MediaOccurrence element is to provide access to the same information as the Segment description scheme, but without the hierarchy and without extra temporal and spatial information. There are some applications for which this information – location of the media, the temporal and/or spatial localization in the media and the audio and visual description tools at that location – is sufficient. If the description requires more information or access to the media, it should use the semantic relations such as *depicts*, *symbolizes*, or *references* instead of the MediaOccurrence element to complement the semantic description. Three types of semantic entity occurrences are considered: a semantic entity can be perceived, symbolized, or referenced (the subject of) in the media.

The AbstractionLevel data type in the SemanticBase description scheme describes the kind of abstraction that has been performed in the description of a semantic entity. When it is not present in the description of a semantic entity, the description is concrete – it describes the world depicted by a specific instance of multimedia content and references the multimedia content (through the MediaOccurrence element or the segment entity tools). If the AbstractionLevel element is present in the description, a media abstraction or a formal abstraction is in place. If the description contains a media abstraction, then the dimension attribute of the AbstractionLevel element is set to zero. If a formal abstraction is present, the dimension attribute is set to one or higher. A semantic description that abstracts specific semantic entities gets an AbstractionLevel element of dimension attribute one. Higher values of the dimension attribute of the AbstractionLevel element are used for abstractions of abstractions, such as structural abstractions in which the elements of a graph are not variables but classes of variables. In this case, the graph relations are the only relevant parts of the description. Examples of concrete descriptions, media abstractions and formal abstractions are discussed in Section 8.4.1.

Finally, the Extent and Position data types are used in the SemanticPlace and SemanticTime description schemes to describe semantic measurements in space or time, respectively. The Extent data type describes the size or extent of an entity with respect to a measurement type (e.g. '4 miles' and '3 weeks'). The Position data type describes the position in space or time of an entity in semantic terms by indicating the distance and direction or angle from an origin (e.g. '4 miles from New York City' and 'The third day of April').

8.4.4 Semantic Relations

The semantic relation tools describe semantic relations. MPEG-7 has standardized common semantic relations such as *agent* and *time* but it allows the description of non-normative relations. Figure 8.1 shows examples of several semantic relationships between the objects, the event and the concept in the description. The normative semantic relations in MPEG-7 are listed in Table 8.2. As for structural relations, MPEG-7 has also standardized the inverse of each semantic relation. Figures 8.1, 8.14 and 8.15 illustrate examples of some of these relations.

The semantic relation tools include the SemanticRelation CS, which specifies semantic relations that apply to entities that have semantic information, which may be implicitly or explicitly stated. Normative semantic relations describe, among others, how

Table 8.2 Normative semantic relations in MPEG-7 listed by type

Type	Normative relations
Semantic	agent, agentOf, patient, patientOf, experiencer, experiencerOf, stimulus, stimulusOf, causer, causerOf, goal, goalOf, beneficiary, beneficiaryOf, them, themOf, result, resultOf, instrument, instrumentOf, accompanier, accompanierOf, summarizes, summarizedBy, state, stateOf
	combination, specializes, generalizes, similar, opposite, exemplifies, exemplifiedBy, interchangeable, identifier, part, partOf, contrasts, property, propertyOf, user, userOf, component, componentOf, substance, substanceOf, entailment, entailmentOf, manner, mannerOf, influences, dependsOn, membershipFunction
	key, keyFor, annotes, annotatedBy, shows, appearsIn, reference, referenceOf, quality, qualityOf, symbolizes, symbolizedBy, location, locationOf, source, sourceOf, destination, destinationOf, path, pathOf, time, timeOf, depicts, depictedBy, represents, representedBy, context, contextFor, interprets, interpretatedBy



Figure 8.15 Picture of a Maya vessel whose semantic description is shown in Figure 8.16 [7]

semantic entities relate in a narrative, how the definitions of semantic entities relate and the localization of semantic entities in space, time and media. As shown in Figure 8.12, the normative semantic relations may apply to entities such as semantic entities (see Section 8.4.2), collections (see [2]), models (see [2]) and segments (see Section 8.3.1), among others.

Normative semantic relations may describe how several semantic entities relate in a narrative or story (e.g. *agent*, *patient* and *accompanier*). For example, semantic relations may describe relations between objects and/or events that draw upon typical linguistic relations such as an object being the agent, patient, experiencer, accompanier, stimulus, cause, goal, or instrument of an event, or an event being the result or a summary of another event, among others.

Other normative semantic relations may describe how the definitions of several semantic entities relate to each other (e.g. *combination*, *specializes* and *exemplifies*). As examples, semantic relations may describe a semantic entity being the combination of the meanings of two or more semantic entities or a semantic entity being a specialization, similar in meaning to, opposite in meaning to, or an example of another semantic entity, among others.

There are semantic entities that may also describe the spatial, temporal, or media localization of semantic entities in segments, models, collections, or other semantic entities (e.g. *depicts*, *symbolizes* and *context*). For example, semantic relations may describe an event taking place at a specific place and time; an object being perceived, referenced, or symbolized in a segment or collection; or a narrative world being the context or interpretation of another narrative world, among others.

Finally, the relations specified by the GraphRelation CS (e.g. *equivalent* [1]), the BaseRelation description scheme (e.g. *member* and *refines* [1]), the SpatialRelation CS (e.g. *above* and *left*, see Section 8.3.4) and the TemporalRelation CS (e.g. *precedes* and *during*, see Section 8.3.4) can also be used to describe relations among semantic entities.

To illustrate the use of the semantic entity and relation tools, consider the example in Figure 8.15 and Figure 8.16. This example shows the picture of a Maya vessel and its semantic description [7]. The textual description of the example in Figure 8.16 could be as follows: 'Vessel created in Guatemala in the 8th Century that is an example of Maya art. The vessel is 14 cm tall and it has several paintings. The paintings show the realm of the lords of death with a death figure that dances and another figure that holds an axe and a hand stone. The paintings represent sacrifice.' The semantic description comprises several semantic entities described using description schemes derived from the SemanticBase description scheme as indicated within parenthesis in Figure 8.16. Several relationships among the semantic entities are also described and represented as named arrows in Figure 8.16. The object 'Vessel' is the *patient of* the event 'Create', whose *location* and *time* are 'Guatemala' and '8th Century', respectively. The object 'Vessel' is 14 cm tall, which is described using the SemanticState description scheme. The object 'Vessel' *exemplifies* the object 'Maya art'. One of the *components of* the object 'Vessel' is the object 'Paintings', which *depicts* the narrative world 'Paintings world'. The *location of* this narrative world is the place 'realm', which is also the *location of* the object 'Lords of Death'. In the narrative world 'Paintings world', agent object 'Death figure' is the *agent of* the event 'Dance' and the event 'Hold' has the agent object 'Figure' as an *agent* and

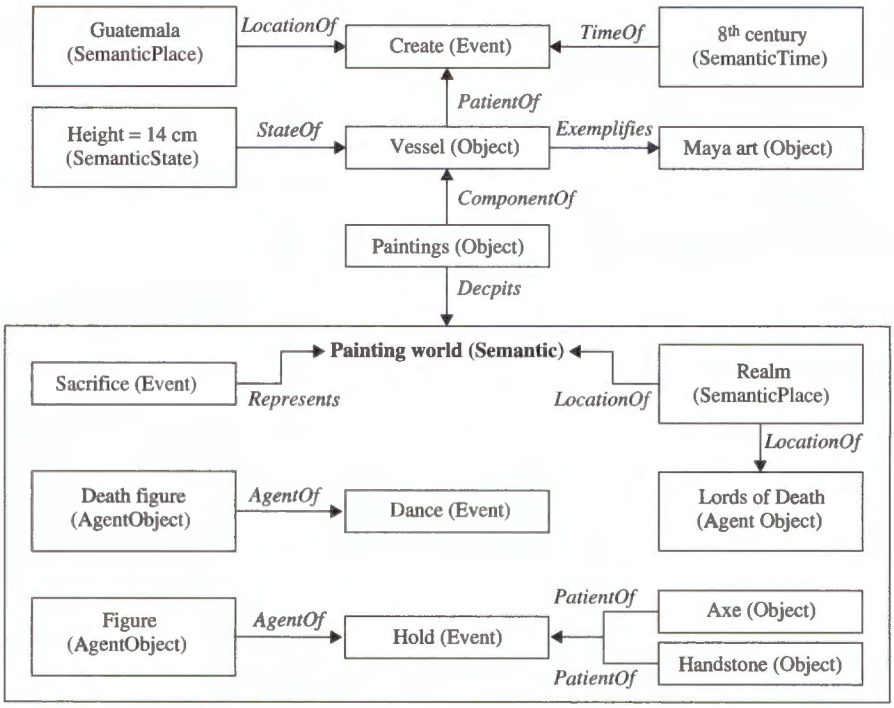


Figure 8.16 Semantic description for the picture shown in Figure 8.15 including several descriptions of semantic entities and relationships among them

the objects 'Axe' and 'Handstone' as *patients*. The narrative world 'Paintings world' is *represented* by the event 'Sacrifice'. All the relationships between semantic entities are normative.

8.4.5 Implicit and Explicit Semantic Description

The content semantic tools provide methods to create very brief or quite extensive semantic descriptions. Because certain relationships are considered more common than others, brevity in semantic description can be achieved by using implicit methods provided by these tools. By contrast, when it is desired to fully describe the interrelation of semantic entities occurring in the media, explicit methods may be used. There are three choices between implicit and explicit methods in creating MPEG-7 semantic descriptions. The first choice is the use of embedded relation tools in the SemanticBase description scheme versus graph tools in the Semantic description scheme. The second choice is the use of the Property element versus the Concept description scheme. The third choice is the use of recursion versus relation tools. These three choices shall be described in this section.

Relation tools in MPEG-7 are quite flexible. From within a particular semantic entity description, say an instance of the Object description scheme, any number of relations may be specified by giving the names of these relations and their targets. In this case, the source of the relation is assumed to be the object being described. One can also

describe a collection of the relations among the semantic entities in a narrative world using the Graph description scheme in the Semantic description scheme. Graph descriptions consist of lists of relation descriptions or of node and relation descriptions. The most explicit way of describing relations is to create an instance of the Graph description scheme with Node elements that reference the entities in the narrative world and the relations among them.

Properties may be described explicitly or implicitly. Implicit properties are described by using the Property element in the SemanticBase description scheme. The relation between such properties and the entity being described is automatically *propertyOf*. In order to indicate a more specific relationship, or allow multiple entities to be related to a single property, or specify the strength of a property, the property or group of properties must be described explicitly. This is done by using the Concept description scheme. Note that this also allows the property to be described as a semantic entity in its own right, with labels, a definition and relationships to other semantic entities.

Recursion allows another method for implicit relationships. Instances of both the Object and Event description schemes may contain other instances of the Objects and Event description schemes, respectively. The relation here is one of composition. It does not specify how the objects and events are composed or the relation between siblings in the decomposition. To do this, relations need to be made explicit, using the Relation or Graph description schemes. But for restricted domains, where these relations are well known, this recursion offers a shorthand for a semantic tree, completely complementary to the trees created by the segment entity tools.

Another implicit relation, the recursion of the Semantic description scheme and, for that matter, for any description scheme derived from the SemanticBag description scheme, deserves special mention. Although this may be represented without making the parent or sibling relations explicit, the meaning cannot be presumed, except by external semantics provided in the application domain. For this reason, this method of recursion usually will require explicit relations. An instance of the Semantic description scheme, for instance, is a description of a single narrative world. If it contains other instances of the Semantic description scheme, these must be either narrative worlds contained in the parent, contextual information, or information used in anaphoric (metaphor, simile, or analogy) description. Where there is an ambiguity that is not clarified by the current semantics, the role of the embedded instance of the Semantic description scheme must be made clear. An example of new semantics would be to derive 'Thesaurus' from the SemanticBag description scheme and specify that the instances of the Semantic description scheme contained in the Thesaurus are examples of complex media terms.

We illustrate the concepts by showing four ways to describe the relationships within the narrative world 'Painting world' in Figure 8.16 (the XML is in the chapter appendix included in the DVD accompanying this book). The first is to explicitly collect together the description of the relationships between the semantic entities, shown in Figure 8.16, into a graph description. In the second, we explicitly list the node descriptions of the graph; such an addition might be used if the graph description were to be used often with graph matching software. In the third, we do a full-blown, fully explicit description of the graph. The nodes are matched to the semantic entities of the narrative world 'Painting world', the relations are between the nodes in the graph, the relationships have associated

identifiers so that the graph may be referenced. The fourth example shows how such a graph can be abstracted. In this case, the graph is not particularly general (unless all Maya paintings depict the same scene, which might be the case for a ritual object), but its elements may then be filled in or matched to any descriptions of the objects, events and people populating the scene.

8.5 CONCLUSION

MPEG-7 provides a comprehensive suite of tools for describing individual multimedia documents that has the potential to revolutionize current multimedia applications. In particular, these tools can describe the content management, structure and semantics of multimedia content allowing efficient searching, filtering and browsing of multimedia content.

REFERENCES

[1] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, Englewood Cliffs, N.J.
[2] J. Smith and A. B. Benitez, Description of Collection of Audio-Visual content and Models, Chapter 10.
[3] P. van Beek and J. Smith, Navigation and Access, Chapter 9.
[4] P. van Beek, K. Yoon and A. M. Ferman, User Interaction, Chapter 11.
[5] J. F. Allen, "Maintaining knowledge about temporal intervals," *Communications of ACM*, **26**(11), 832–843 (1983).
[6] *http://www.getty.edu/research/institutestandards/cdwa/3_cataloging_examples/index.html, Accessed on 6/1/2001*.
[7] Getty Research Institute, Getty Standards Program, Categories for the Description of Works of Art. J. Paul Getty Trust and the College Art Association (2001). Cataloging Examples: Maya Vessel.
[8] T. Walker, J. Heuer and J. M. Martinez, MPEG-7 Basic MDS elements, Chapter 7.

9

Navigation and Summarization

Peter van Beek[1] and John R. Smith[2]

[1]*Sharp Laboratories of America, Camas, USA,* [2]*IBM T. J. Watson Research Center, Hawthorne, USA*

9.1 INTRODUCTION

Digital audiovisual material is becoming increasingly available to users through digital TV broadcast, digital photo and video cameras, DVD and PC-based access to movie clips and music songs on the Internet. Standards and tools for efficient audio and video compression, such as MPEG-1, 2 and 4, are accelerating this trend, as is the increasing availability of broadband Internet connections, and high-capacity, hard disk–based storage allowing nonlinear access to audiovisual content. Furthermore, the content access environment is becoming more heterogeneous, seeing a significant increase in wireless networks and mobile devices being used.

Consequently, there is a need for rapid navigation and browsing capabilities and tools for flexible access, to enable users to efficiently discover and consume multimedia content. Users will also benefit from having access to multiple views and variations of content. These views and variations can be adapted or selected on the basis of the user's personal preferences, interests or usage conditions, such as the amount of time the user wants to spend in consuming the content or the resources available to the user's terminal. Such adaptability will significantly enhance the usability of multimedia information.

This chapter describes navigation and summarization concepts, the relevant MPEG-7 tools and examples of their usage (also see [1, 2]). We discuss Description Schemes that support various types of summaries and views of multimedia content. Firstly, we discuss *summarization* tools, which are used primarily to specify summaries of time-varying audiovisual data that support sequential and hierarchical navigation. Examples of such summaries are audiovisual highlights, visual table of contents, audiovisual previews and slide shows. Secondly, we discuss *view* tools, which are used to specify different partitions and decompositions of image, video and audio signals in space, time and frequency and support progressive retrieval of the data. Examples of partitions are spatial regions of images, temporal-frequency subbands of audio and 3-D wavelet subbands of video. Finally, we discuss tools that relate multiple, different, *variations* of multimedia

programs, in which variations may result from various types of processing, such as summarization, translation or transcoding.

The tools standardized by MPEG-7 will support effective content retrieval, navigation and browsing. Furthermore, the MPEG-7 navigation tools satisfy the following requirements.

- Simplicity, modularity and extensibility. Simple enough to be used on a broad range of platforms, while extensible to many different applications.
- Independence of transport or delivery mechanism. Allow transport via broadcast, Internet, stored and streamed and is independent of media format.
- Independence of rendering or presentation engine. Not constrained to particular hardware platforms or browsing or viewing applications.
- Support sequential and hierarchical browsing and navigation. Enable audiovisual summaries, audiovisual table of contents, segment jumping. Provide access at progressive levels of detail. Enable bookmarking of segments or key frames by the user and exchanging bookmarks between users.
- Support automatic adaptation and personalization of multimedia content to terminal capabilities and user's preferences. Support universal access to multimedia content.

9.2 SUMMARIZATION

This section describes MPEG-7 concepts and tools related to summarization [3, 4]. The MPEG-7 summarization tools are based on the following key concepts.

- *Original (source) content*: The original content that is being summarized, parts of which may be used in a summary. There may be one or more source content items being used in a single summary. The original content can be identified using one or more unique identifiers and can be located using one or more media locators (e.g. a URL and/or time stamps).
- *Summary content*: Parts or components of the original (source) content that are used to create a summary. Examples of such components are key video segments, key audio segments and key frames. These parts of the original content are located using timestamp elements defined with respect to the time-base of the original content. Note that a summary may also use *alternative* or *external content* that is not physically part of the original, to represent parts of the original. Such content may be located using media locators (e.g. a URL).
- *Summary*: An abstract, extract or digest of the original (source) content that conveys the essential information of the original. A summary is a composite of the summary content that can be presented to the user. For example, a summary may simply be a concatenation of video segments taken directly from the original, in which the segments are specified by their start time and duration. A summary may have its own time-base, which allows for presenting summary content, for example, audio segments, at increased or decreased speeds.

A MPEG-7 summary description specifies the summary content, how it relates to the original content and how a summary can be composed from these and presented to the user. Note that summary descriptions may be part of the same overall description that describes the original content. Within MPEG-7, two description schemes in particular have been developed in the area of summarization: the *HierarchicalSummary* and *SequentialSummary* description schemes. The HierarchicalSummary description scheme is used to specify summaries of time-varying audiovisual (AV) data that support sequential and hierarchical navigation. These summaries are composed of audio, video or AV segments, as well as their key frames and key sounds. For example, a highlight summary of television content can be composed using the HierarchicalSummary description scheme. The SequentialSummary description scheme is used to specify summaries of time-varying AV data that support sequential navigation. These summaries are composed of images or video frames and audio clips, which can be synchronized and presented to the user at varying speeds. For example, a variable-speed fast-forward of a video or an audio slide show can be composed using the SequentialSummary description scheme.

The models underlying these description schemes are illustrated in Figures 9.1 and 9.2, using notation based on the Unified Modeling Language (UML) (see [5]). Each rectangular box corresponds to a description scheme or descriptor; paths between boxes denote composition relationships and strings such as '1,*' indicate the multiplicity of the relationship.

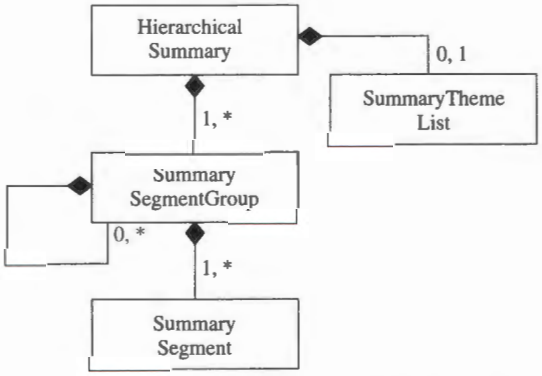


Figure 9.1 HierarchicalSummary description scheme

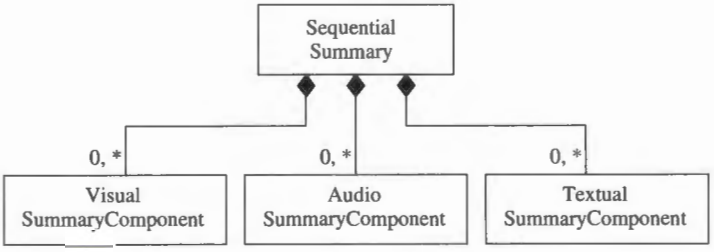


Figure 9.2 SequentialSummary description scheme

Both a HierarchicalSummary and a SequentialSummary may contain: a *Name* attribute, to name a summary; a *SourceID* attribute, to identify the original content; a *SourceLocator* attribute, to locate the original content; and a *SourceInformation* attribute, to reference elements of the description of the original content.

A *HierarchicalSummary* contains one or more summaries, each represented by a *SummarySegmentGroup*. Each summary provides an alternative view of the same AV content. A HierarchicalSummary may also contain a *SummaryThemeList*, which is a list of the names of themes or events that may be referenced by other summary components.

A *SummarySegmentGroup* contains one or more *SummarySegments,* which correspond to basic segments of the AV content. A SummarySegmentGroup may also contain one or more children SummarySegmentGroup elements, which represent the summary at a finer level of detail. This construct may be used to build hierarchical summaries.

A *SummarySegment* contains time stamps indicating the location of the audio and/or video data associated with this segment. A SummarySegment may also contain time stamps that locate key frames and key sounds. A SummarySegment also has an *order* attribute to specify the navigation or presentation order of segments in a summary. Note that each SummarySegment element may refer to a different source using a separate *SourceID* element. Both a SummarySegmentGroup and a SummarySegment may contain a *Name* attribute and *references* to summary themes or events, to describe a summary or a segment in textual form.

The following is an abbreviated example of a hierarchical summary in Extensible Markup Language (XML) format. This summary description is constructed with two levels, where the base level (level 0) is defined by a group of three key AV clips, and the next level (level 1) contains a group of two additional key AV clips. The three clips included in the base level can be used to form a coarse video summary. The two clips included in the next level can be added to the clips in the base level to form a more detailed video summary. Note the use of the order attribute to indicate the presentation order of the AV clips. This summary contains segments showing particular events in a video program, in particular 'slam dunks', 'three-point shots' and 'two-point shots.' These event names are listed in the SummaryThemeList element. Note that each such event or theme name may be referenced multiple times, and each segment may refer to multiple themes or events. The source content being summarized is identified by a Universal Resource Name (URN) and the summary data (segments) is taken directly from the source data. Note that some of the timing information in this example was replaced by XML comments, for brevity.

```
<HierarchicalSummary hierarchy="dependent">
 <SourceID>urn:mymedia:av:v00050</SourceID>
 <SummaryThemeList>                    <!--themes or events-->
   <SummaryTheme id="E0"> slam dunk </SummaryTheme>
   <SummaryTheme id="E1"> 3-point shot </SummaryTheme>
   <SummaryTheme id="E2"> 2-point shot </SummaryTheme>
 </SummaryThemeList>
 <SummarySegmentGroup level="0">       <!--level 0 segment group-->
   <Name>Game Highlights</Name>
   <SummarySegment themeIds="E1" order="0">
     <KeyAudioVisualClip>              <!--key clip #0.1-->
       <!--start time and duration-->
```

```
      </KeyAudioVisualClip>
    </SummarySegment>
    <SummarySegment themeIds="E0" order="2">
      <KeyAudioVisualClip>            <!--key clip #0.2-->
        <!--start time and duration-->
      </KeyAudioVisualClip>
    </SummarySegment>
    <SummarySegment themeIds="E0 E1" order="3">
      <KeyAudioVisualClip>            <!--key clip #0.3-->
        <!--start time and duration-->
      </KeyAudioVisualClip>
    </SummarySegment>
    <SummarySegmentGroup level="1">      <!--level 1 segment group-->
      <SummarySegment themeIds="E2" order="1">
        <KeyAudioVisualClip>             <!--key clip #1.1-->
          <!--start time and duration-->
        </KeyAudioVisualClip>
      </SummarySegment>
      <SummarySegment themeIds="E2" order="4">
        <KeyAudioVisualClip>            <!--key clip #1.2-->
          <!--start time and duration-->
        </KeyAudioVisualClip>
      </SummarySegment>
    </SummarySegmentGroup>
  </SummarySegmentGroup>
</HierarchicalSummary>
```

A *SequentialSummary* consists of three 'tracks' of elements, describing the video and audio components of time-varying AV data and associated textual information (see Figure 9.2). A Sequential Summary may contain: zero or more *VisualSummaryComponent* elements, each describing a single video frame; zero or more *AudioSummaryComponent* elements, each describing an audio clip; and zero or more *TextualSummaryComponent* elements, each describing textual annotation. A SequentialSummary may also contain locators to a precomposed audio, visual or audiovisual summary.

A *VisualSummaryComponent* element contains a *VideoSourceLocator* to locate the source data, and a *ComponentSourceTime* element to locate a particular frame within the original video data. An *ImageLocator* element is used to locate the actual image data used in the summary to represent this frame.

An *AudioSummaryComponent* element contains an *AudioSourceLocator* to locate the source data, and a *ComponentSourceTime* element to locate a particular audio clip within the original audio data. A *SoundLocator* element is used to locate the actual audio data used in the summary to represent the original audio data.

A *TextualSummaryComponent* element contains an *AudioVisualSourceLocator* to locate the source data, and a *ComponentSourceTime* element to locate a particular AV clip within the original data. A *FreeText* element contains the textual information associated with the AV clip.

The various video and audio components can be synchronized across the tracks. To this end, VisualSummaryComponent, AudioSummaryComponent and TextualSummaryComponent elements may contain a *SyncTime* element, describing the start time and

duration of the presentation of that component. Note that each VisualSummaryComponent, AudioSummaryComponent and TextualSummaryComponent element may refer to a different source using separate *SourceID* elements.

The following is a very simple, abbreviated, example of a sequential slide-show summary in XML format. This summary description shows two key frames, to be presented in sequence, along with audio data. The source content being summarized is identified by a URN; however, the summary data (key frames) is stored in separate Joint Photographic Experts Group (JPEG) files.

```
<SequentialSummary>
 <SourceID>urn:mymedia:av:v00038</SourceID>
 <AudioSummaryLocator>                <!--audio summary-->
   <MediaUri> http://www.mymedia.com/av/a1.mp3</MediaUri>
 </AudioSummaryLocator>
 <VisualSummaryComponent>   <!--key frame #1-->
   <ImageLocator>             <!--location of image data-->
     <MediaUri> http://www.mymedia.com/av/f1.jpg</MediaUri>
   </ImageLocator>
   <SyncTime>
     <!-- start time and duration e.g. 0-10 seconds-->
   </SyncTime>
 </VisualSummaryComponent>
 <VisualSummaryComponent>   <!--key frame #2-->
   <ImageLocator>             <!--location of image data-->
     <MediaUri> http://www.mymedia.com/av/f2.jpg</MediaUri>
   </ImageLocator>
   <SyncTime>
     <!--start time and duration e.g. 10-15 seconds-->
   </SyncTime>
 </VisualSummaryComponent>
</SequentialSummary>
```

9.3 VIEWS AND VIEW DECOMPOSITIONS

This section describes MPEG-7 concepts and tools related to space and frequency views and decompositions of image, video and audio signals (see [3, 6]). The MPEG-7 space and frequency tools are based on the following key concepts, illustrated in Figure 9.3.

- *Signal*: Continuous or discrete-sample data in space and/or time. Example signals include discrete 1-D audio signals, 2-D discrete spatial images or time-varying video signals.
- *View*: Region of an image, video or audio signal in multidimensional space-time and/or frequency, which is defined in terms of a space and frequency partition. A *Partition* is a continuous multidimensional rectangular region defined in the space-time and/or frequency plane. The image, video or audio signal is referred to as the Source of the View, while the View signal is referred to as the Target of the View.
- *View decomposition*: Organized set of views that provides a structured decomposition of an image, video or audio signal in multidimensional space and/or frequency. A decomposition may be a *View Set*, a *View Tree*, or a *View Graph*.

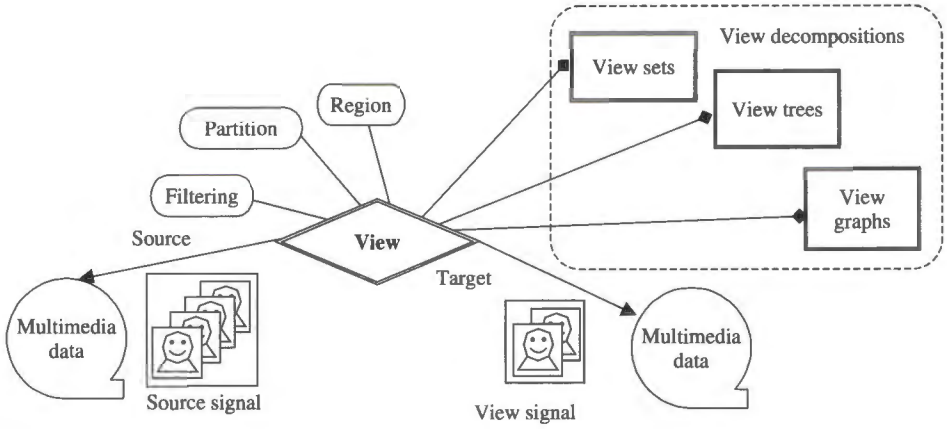


Figure 9.3 Illustration of key concepts of space and frequency views, partitions and decompositions including views, ViewSets and View Decompositions

The different types of Views have properties that denote the specific partitioning in the space and/or frequency plane, the filtering information for extracting the View from the frequency plane and information for locating the regions or segments in space or time relative to the Source signal. For example, a view can correspond to a spatial region of an image, a temporal-frequency subband of an audio or a 3-D wavelet subband of a video. The different View Decompositions are formed as sets or structured decompositions (forming a tree or graph) of the Views. The ViewSet describes a set of Views, for example, which can be a complete and nonredundant set of subbands of an audio signal, a set of wavelet subbands of an image or an incomplete set of regions of an image. The ViewTree and ViewGraph decompositions describe tree- and graph-structured decompositions of the image, audio and video signals, such as a spatial quad-tree or wavelet decomposition of an image.

Figure 9.4 shows an example of a space and frequency graph view decomposition of an image. The views correspond to partitions of the 2-D image signal in space (spatial segments), frequency (wavelet subbands) and space and frequency (wavelet subbands of spatial segments). The space and frequency graph also contains transitions that correspond to the analysis and synthesis dependencies among the views.

The following example describes a tree decomposition in frequency of an audio signal, with branching factor of two at the root node. The example FrequencyTree shows one frequency view at the first level of the tree (the first View element) and another frequency view at the second level of the tree (the View element contained in the first Child element). Each FrequencyPartition element specifies a partition along the frequency axis associated with the target view.

```
<FrequencyTree branching="2">
 <Source>    <!--Source signal for Views in all tree nodes-->
  <AudioSignal>
   <MediaLocator><MediaUri>audio.mp3</MediaUri></MediaLocator>
  </AudioSignal>
 </Source>
```

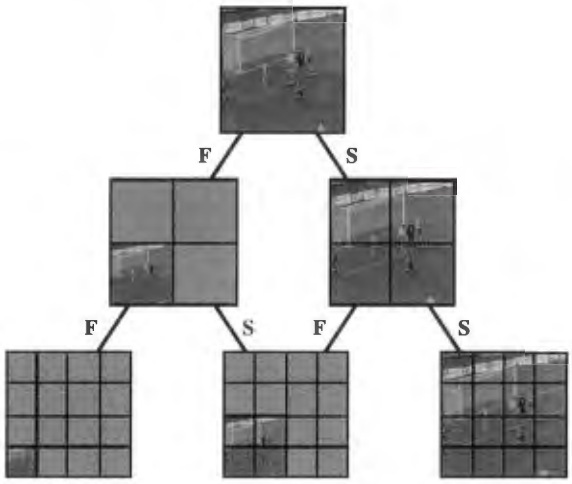


Figure 9.4 Example space and frequency graph view decomposition that describes the decomposition of an image in space and frequency, where 'S' denotes a decomposition in space and 'F' denotes a decomposition in frequency

```
<View>
 <Target>   <!--Target of View in root node-->
  <AudioSignal>
   <MediaLocator><MediaUri>audio_1.mp3</MediaUri></MediaLocator>
  </AudioSignal>
 </Target>
 <FrequencyPartition>
   <Start xsi:type="SignalPlaneFractionType" t="0"/>
   <End xsi:type="SignalPlaneFractionType" t="0.5"/>
 </FrequencyPartition>
</View>
<Child branching="0">       <!--First child node-->
 <View>
  <Target>    <!--Target of View in first child node-->
   <AudioSignal>
<MediaLocator><MediaUri>audio_11.mp3</MediaUri></MediaLocator>
   </AudioSignal>
  </Target>
  <FrequencyPartition>
    <Start xsi:type="SignalPlaneFractionType" t="0"/>
    <End xsi:type="SignalPlaneFractionType" t="0.25"/>
  </FrequencyPartition>
 </View>
</Child>
<Child branching="0">      <!--Second child node-->
 <!--Target and FrequencyPartition in second child node-->
</Child>
</FrequencyTree>
```

9.4 VARIATIONS

This section describes MPEG-7 concepts and tools related to variations of multimedia programs [3, 7]. The MPEG-7 variation tools are based on the following key concepts, which are illustrated in Figure 9.5.

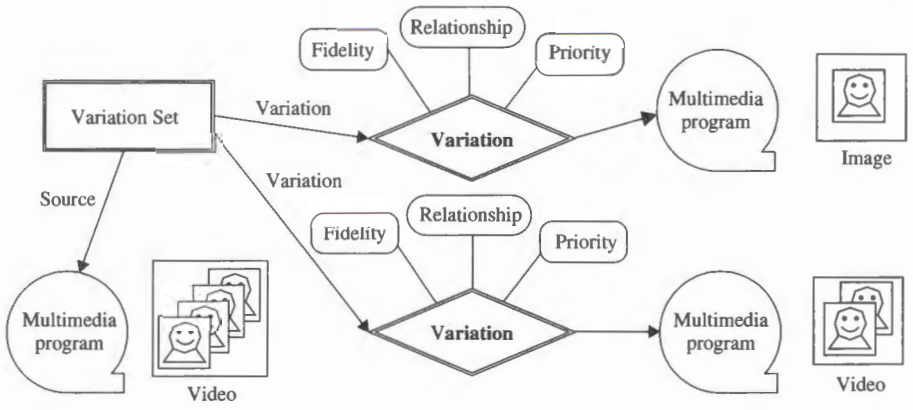


Figure 9.5 Illustration of key concepts of Variation tools including VariationSet, Variation, Variation-Fidelity, Variation-Priority and Variation-Relationship

- *Source program*: A multimedia program that is the source of one or more variations;
- *Variation program*: A multimedia program that is a variation of a source program;
- *Variation set*: A set of variations of a multimedia program.

The variation tools describe different variations of a source multimedia program such as compressed or low-resolution versions, summaries, different language translations and alternative modalities. The variations may refer to newly authored multimedia programs or correspond to multimedia content derived from another source. A variation-fidelity value gives the quality of the variation compared with the original. The variation-relationship attribute indicates the type of variation, such as *summary*, *abstract*, *extract*, *modality translation*, *language translation*, *color reduction*, *spatial reduction*, *rate reduction*, *compression* and so forth. The variation-priority attribute indicates the priority of the variation program with respect to the other programs included in the same variation set.

The following XML example describes a set of variation programs using the VariationSet description scheme. A VariationSet describes first the source AV program by using an AudioVisual element and a MediaLocator element. Then following in the description is a set of three variation programs (labeled B, C and E). For each variation program, the Variation element defines the fidelity, priority and VariationRelationship. The example shows in some cases that multiple VariationRelationships can be specified, for example, the variation program E is derived from the source program via both spatial reduction and compression.

```
<VariationSet>
 <Source xsi:type="AudioVisualType">
  <AudioVisual>
   <MediaLocator>
    <MediaUri>file://soccer-A.mpg</MediaUri></MediaLocator>
  </AudioVisual>
 </Source>
 <Variation xsi:type="ImageType" fidelity="0.85" priority="1">
  <Image>
   <MediaLocator>
    <MediaUri>file://soccer-B.jpg</MediaUri></MediaLocator>
```

```
  </Image>
  <VariationRelationship>extraction
  </VariationRelationship>
 </Variation>
 <Variation xsi:type="AudioType" fidelity="0.75" priority="3">
  <Audio>
   <MediaLocator>
    <MediaUri>file://soccer-C.mp3</MediaUri></MediaLocator>
  </Audio>
  <VariationRelationship>extraction
  </VariationRelationship>
  <VariationRelationship>languageTranslation
  </VariationRelationship>
 </Variation>
 <Variation xsi:type="AudioVisualType" fidelity="0.5"
priority="2">
  <AudioVisual>
   <MediaLocator>
    <MediaUri>file://soccer-E.mpg</MediaUri></MediaLocator>
  </AudioVisual>
  <VariationRelationship>spatialReduction
  </VariationRelationship>
  <VariationRelationship>compression
  </VariationRelationship>
 </Variation>
</VariationSet>
```

9.4.1 Automatic Extraction and Examples of Usage

In the following text, we briefly discuss automatic extraction of descriptions and examples of usage.

9.4.2 Summarization

Efforts toward efficient video browsing and summarization date back to the early nineties [8]. Early research focused on automatic shot boundary detection and key-frame extraction, using low-level cues such as motion and color. However, summaries that contain truly relevant or interesting events (e.g. highlights of sports games and such or news synopses) require more intelligent automatic algorithms. Currently, part of the content description may have to be obtained through a manual process, most likely by a service provider or the content provider. However, several advances have been made in automatic summarization technology recently, such as [9–14]. Some of these recent approaches are directed at detecting specific semantic events [13] or are constrained to particular domains, such as news programs [15] or sports programs [12, 16]. Others exploit human interaction in a nonintrusive manner [11].

The summarization tools presented above support a wide variety of browsing and navigational modes. The HierarchicalSummary tools can be used to implement a *table of contents*, which is a hierarchical set of access points into a multimedia program. *Highlights* are another type of summary, containing a sequence of audiovisual key segments that are to be played back in a specific order. *Synopses* and *previews* are similar to highlights, but have different purposes (highlights are entertaining; synopses are informational; previews

can be used to decide whether to view an entire program). Summaries may also be generated by users by *bookmarking* of segments or access points. Furthermore, a *virtual program* can be constructed by concatenating segments from different original (source) programs. The SequentialSummary tools can be used to create slide shows with key frames accompanied by audio, smart quick views or video skims [9] that adapt to the amount of scene activity and audio summaries.

9.4.3 View Decompositions

By decomposing the image, video and audio signals using spatial and temporalpartitioning and/or filter banks, the view descriptions can be generated automatically. For example, the space and frequency view decomposition shown in Figure 9.4 can be generated by spatially partitioning the image using a spatial quad-tree combined with wavelet tree decomposition of each of the spatial image segments [17].

By describing structural views and decompositions of the signals in the spatial and/or frequency domain, the space and frequency tools can be used to build representations of the signals that enable multiresolution access or progressive retrieval. For example, the space and frequency view decomposition shown in Figure 9.4 has been used for representation and progressive retrieval of large high-resolution images [6]. The view decompositions of video signals in spatial- and temporal-frequency have also been used as a basis for progressive retrieval of video sequences [18]. The view decompositions have also been used for progressive retrieval in which the synthesis processing of the views is adaptively partitioned between server and client according to client cache status and processing capabilities [19].

9.4.4 Variations

The variation tools provide important content management functionalities by tracking the variations of multimedia content that result from various types of multimedia processing such as summarization, translation, reduction, revision, transcoding and so forth. As a result, the variation tools serve an important role in applications such as Universal Multimedia Access by allowing the selection among different variations of multimedia programs, in order to adapt to the specific capabilities of the terminal devices, network conditions or user preferences. For example, the variation tools can be used for representing and selecting alternative variations of multimedia content at a server in order to adapt the delivery to client devices with different capabilities or to other conditions of the user environment [7].

Table 9.1 Summary of the MPEG-7 tools introduced in this chapter

Tool	Use of tool
Summarization	Navigate and summarize multimedia content sequentially or hierarchically
Views and View Decompositions	Represent partitions and hierarchical decompositions of multimedia content, for multiresolution access
Variations	Describe related variations of multimedia content that result from processing the content, for universal access

9.5 CONCLUSION

In this chapter, we have introduced MPEG-7 description schemes for navigating, browsing and retrieval of multimedia content. Table 9.1 provides an overview of these tools.

REFERENCES

[1] P. Salembier, R. Qian, N. O'Connor, P. Correia, I. Sezan and P. van Beek, "Description schemes for video programs, users and devices," *Signal Processing: Image Communication*, **16**, 211–234 (2000).

[2] I. Sezan and P. van Beek, "MPEG-7 standard and its expected role in development of new information appliances," *International Conference on Consumer Electronics 2000*, Los Angeles, Calif., 2000, pp. 274–275.

[3] ISO/IEC 15938, Information Technology – Multimedia Content Description Interface, 2001.

[4] P. van Beek, I. Sezan, D. Ponceleon and A. Amir, "Content description for efficient video navigation, browsing and personalization," *Workshop on Content-based Access of Image and Video Libraries 2000*, IEEE Hilton Head, SC, 2000, pp. 40–44.

[5] S. S. Alhir, *UML in a Nutshell*, O'Reilly & Associates, Sebastopol, Calif., 1998.

[6] J. R. Smith and C.-S. Li, "Adaptive storage and retrieval of large compressed images," *Symposium on Electronic Imaging: Science and Technology – Storage & Retrieval for Image and Video Databases VII*, San Jose, Calif., 1999, pp. 467–479.

[7] J. R. Smith, R. Mohan, C.-S. Li, "Scalable multimedia delivery for pervasive computing," *International Conference on Multimedia*, Part 1, Orlando, Fla., 1999, pp. 131–140.

[8] H. Zhang, C. Y. Low and S. W. Smoliar, "Video parsing and browsing using compressed data," in *Multimedia Tools and Applications*," Vol. 1, Kluwer Academic Publishers, Norwell, Mass., 1995.

[9] M. G. Christel, M. A. Smith, C. R. Taylor and D. B. Winkler, "Evolving video skims into useful multimedia abstractions," *Proceedings of the ACM Computer-Human Interface Conference 1998*, Los Angeles, Calif., 1998, pp. 117–178.

[10] A. Hanjalic and H. Zhang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," *IEEE Transaction on Circuits and Systems for Video Technology*, **9**, 1280–1289 (1999).

[11] R. Lienhart, "Dynamic video summarization of home video," *Storage and Retrieval for Media Databases 2000*, Proceedings of SPIE, Vol. 3972, San Jose, Calif., 2000, pp. 378–391.

[12] H. Pan, P. van Beek and I. sSezan, "Detection of slow-motion replay segments in sports video for highlights generation," *International Conference on Acoustics, Speech and Signal Processing 2001*, Salt Lake City, Utah, 2001, pp. 1649–1652.

[13] R. Qian, N. Haering and I. Sezan, "A computational approach to semantic event detection," *International Conference on Computer Vision and Pattern Recognition 1999*, Vol. 1, Fort Collins, Colo., 1999, pp. 200–206.

[14] D. M. Russell, "A design pattern-based video summarization technique," *Proceedings of the 33rd Hawaii International Conference on System Sciences – 2000*, Maui, Hawaii, 2000, pp. 1137–1141.

[15] M. T. Maybury and A. E. Merlino, "Multimedia summaries of broadcast news," *Proceedings of the 1997 IASTED International Conference on Intelligent Information Systems*, Grand Bahama Island, Bahamas, 1997, pp. 442–449.

[16] Y. Rui, A. Gupta and A. Acero, "Automatically extracting highlights for TV baseball programs," *Proceedings of ACM Multimedia 2000*, Los Angeles, Calif., 2000, pp. 105–115.

[17] J. R. Smith and S.-F. Chang, "Joint adaptive space and frequency graph basis selection," *International Conference on Image Processing 1997*, Santa Barbara, Calif., 1997, pp. 702–705.

[18] J. R. Smith, "VideoZoom spatio-temporal video browser," *IEEE Transaction Multimedia*, **1**(2), 157–171 (1999).

[19] J. R. Smith and C.-S. Li, "Adaptive synthesis in progressive retrieval of audio-visual data," *International Conference on Multimedia and Expo 2000*, New York, 2000, pp. 339–342.

10

Content Organization

John R. Smith[1] and Ana B. Benitez[2]

[1]*IBM T. J. Watson Research Center, Hawthorne, USA,* [2]*Columbia University, New York, USA*

10.1 INTRODUCTION

With the tremendous growth of digital multimedia data, multimedia systems and technologies are evolving rapidly to meet this challenge. The evolution can be described as one in which the technology focus is moving from treating multimedia content as signals to more advanced extraction and processing of multimedia features, semantics and knowledge, as illustrated in Figure 10.1. By initially handling multimedia content as signals, the MPEG-1 and MPEG-2 audiovisual coding standards have allowed efficient storage, compression and communication. More recently, by extracting and analyzing features of the multimedia signals, MPEG-4 has further improved coding efficiency by using, for example, object-based coding techniques. In addition, MPEG-7 allows content-based retrieval of multimedia by extracting and searching features using similarity measures. However, the challenge remains for automatically extracting semantic labels of multimedia content, including labeling of objects, events, places, people and so forth. By labeling multimedia content at the semantic level, the content will be easier to search, filter, index, summarize, personalize and repurpose. Ultimately, semantic labels allow the extraction of knowledge through the mining of multimedia archives. This can enable business use of multimedia and decision support and allow seamless marketplaces for multimedia content.

In bridging these gaps from multimedia signals to knowledge, models have an important role. Models provide parameterized or statistical representations of the multimedia information. Given a collection of digital images of sunsets, the color features of the images can be used for forming a model of sunset images. This can be accomplished by extracting color histogram descriptors from the images and by computing a centroid color histogram feature vector. By modeling features and semantic information in this way, unknown images and, in general, multimedia content can be labeled by fitting its features to the models, or equivalently, by classifying the content according to the models. The color histogram model can be used to classify unknown images by comparing a color histogram feature vector of an unknown image with the centroid color histogram feature vector of the sunset image semantic class.

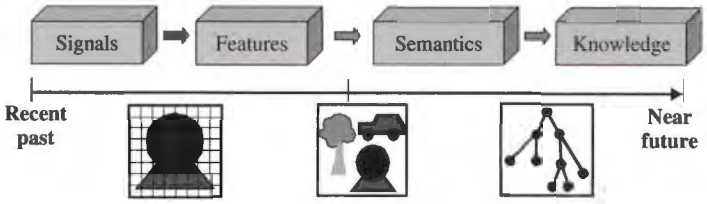


Figure 10.1 Evolution of multimedia applications and technologies involves transition from dealing with signals to features to semantics to knowledge. Models help in bridging the gap from signals to knowledge

Given the importance of semantic labels and the need for modeling content at the feature level in order to automatically annotate and classify multimedia content, MPEG-7 has standardized tools for describing collections and models as follows:

- *Collection tools*: The collection tools describe collections of multimedia content, segments, descriptor instances, concepts or mixed content. Collections can be used for the tasks of describing an album of songs, a group of objects or a cluster of color feature descriptors. The model tools describe parameterized models of multimedia content or collections.
- *Model tools*: The model tools include probability models, analytic models and cluster models. The probability models describe statistics or probabilities associated with the attributes of multimedia content, descriptors or collections. The analytic models describe the association of labels or semantics with multimedia content or collections. The cluster models further describe the association of probability models with the analytic models. Cluster models can describe the centroid of a collection of color feature descriptions of a semantic class of images. Classification models are composed of sets of cluster models or analytic models, and describe information that can be used for classifying multimedia content, as described later.

The MPEG-7 content organization tools are based on the following key concepts:

- *Collection*: Unordered sets of multimedia content, segments, descriptors, concepts or mixed sets of the above.
- *Model*: Parameterized representation of an instance or class multimedia content, descriptors or collections, as follows:
  - *Probability model*: Associates statistics or probabilities with the attributes of multimedia content, descriptors or collections.
  - *Analytic model*: Associates labels or semantics with multimedia content or collections.
  - *Cluster model*: Associates labels or semantics and statistics or probabilities with multimedia content collections.
  - *Classification model*: Describes information about known collections of multimedia content in terms of labels, semantics and models that can be used to classify unknown multimedia content.

10.2 COLLECTIONS

This section describes MPEG-7 concepts and tools related to collections [1]. As shown in Figure 10.2, the Collection tools include tools for content collections, segment collections, descriptor collections, concept collections, mixed collections and structured collection. The Content Collection tool describes collections of multimedia documents, whereas the Segment Collection tool describes collections of segments of multimedia data, including

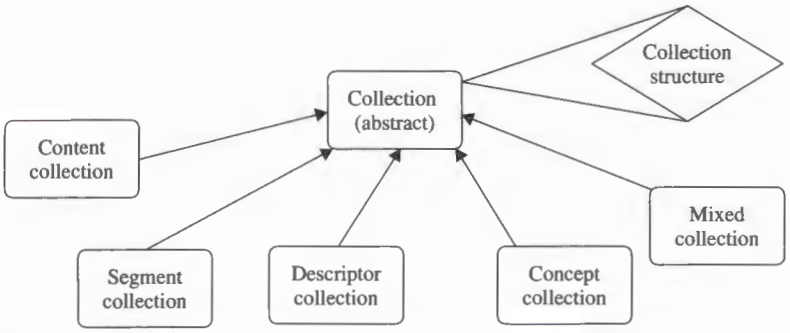


Figure 10.2 Overview of Collection tools – Content Collection, Segment Collection, Descriptor Collection, Concept Collection and Mixed Collection tools are specialized types of Collection tools. The Structured Collection describes the relationships among Collections

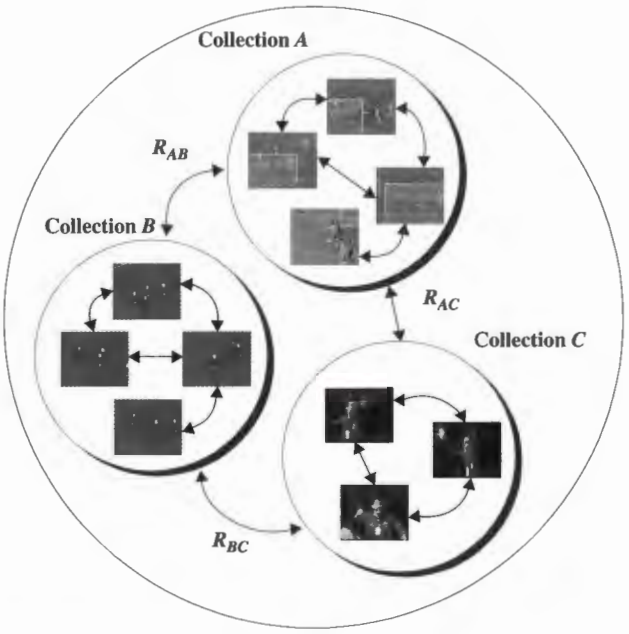


Figure 10.3 The Content Collection tools describe a collection of multimedia content. Collection Structure tool describes the relationships across multiple collection

image regions and video segments. The Descriptor Collection tool describes collections of multimedia content descriptions that are instances of descriptors, such as the MPEG-7 Dominant Color Descriptor (DCD) or Scalable Color Descriptor. The Concept Collection tool describes collections of semantic concepts, such as objects, events, people and places. The Mixed Collection tool describes collections of content, descriptions and concepts. Finally, the Structured Collection tool describes the relationships among collections.

As described above, the Collection tools describe collections related to multimedia content. Figure 10.3 shows an example of a collection structure including multiple collections of images. In this example, each collection consists of a set of images, perhaps with common properties, each depicting similar events in a soccer game. Within each collection, the relationships among the images can be described, such as the degree of similarity of the images in the cluster or spatiotemporal relationships. Across the collections, the degree of similarity of the collections can be described.

10.3 MODELS

This section describes MPEG-7 concepts and tools related to models [1]. The Models are expressed in terms of statistics or probabilities associated with the attributes of collections of multimedia content or are expressed through examples or exemplars of the multimedia content classes. The Model tools describe parameterized models of multimedia content, descriptors or collections. Figure 10.4 shows an overview of the different types of models: probability models, analytic models, cluster models and classification models. Each of these types of Model tools is further broken down into specific types of Model tools. The Probability Model tool describes different statistical functions and probabilistic structures, which can be used to describe samples of multimedia content and classes of descriptors using statistical approximation. The Analytic Model tool describes a collection of examples of multimedia content or clusters of descriptors that are used to provide a model for a particular semantic class. A collection of art images labeled with tag indicating that the paintings are examples of the Impressionist period forms an analytic model. The Cluster Model tool describes analytic models and their associated probability models. This allows the description of the centroid of a collection of descriptor instances

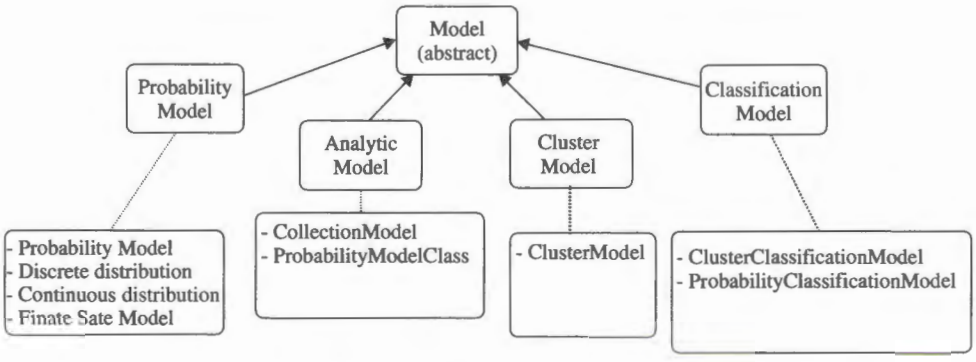


Figure 10.4 Overview of the different types of models: probability models, analytic models, cluster models and classification models

in order to represent a semantic class. Finally, the Classification Model tool describes probabilistic information that can be used for classifying multimedia.

10.3.1 Probability Models

This section describes MPEG-7 concepts and tools related to probability models [1]. The Probability Model tool describes attributes and features of multimedia content using probabilities and statistics and apply to different types of discrete probability distributions, continuous probability distributions and finite-state models.

A probability distribution characterizes the statistical properties of random variables in the form of a distribution. For many purposes, pieces of information suffice for characterizing the distribution. Information about the location (mean) and dispersion (variance) is most frequently used. In general, distributions are discrete or continuous. In the discrete case, a probability mass function is described by specifying its form and parameters (i.e. probability of success, number of trials and so forth). Furthermore, an arbitrary discrete probability mass function is described by specifying the values of the distribution directly using histograms. In the continuous case, a probability density function is likewise described by specifying its form and parameters (i.e. mean, variance, moments, shape and so forth).

Tools for describing probability distributions are provided that cover cases of both discrete and continuous probability distributions. The Probability Distribution tool includes attributes that are common to all distributions, including mean, variance, minimum, maximum, mode, median, moment and so forth. In general, the probability distributions are multidimensional and are described using parameters of the distributions that are vectors. In the case that a distribution is one-dimensional, only the first element of each vector is used, which specifies the corresponding parameter as a scalar quantity. The different forms of discrete probability distributions that are described by the discrete distribution tools include arbitrary histograms and binomial, hypergeometric, Poisson, geometric and uniform distribution types. The different forms of continuous probability distributions that are described by the continuous distribution tools include Gaussian, generalized Gaussian, exponential, gamma, continuous, lognormal and Gaussian mixture model distribution types. For example, the probability density function for a generalized Gaussian distribution given as follows:

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{|(x-\mu)|^r}{2\sigma^2}\right],$$

for $-\infty < x < \infty$ and $\sigma^2 > 0$, where μ corresponds to the mean, σ corresponds to the standard deviation and r corresponds to the shape, is described by specifying the values for the mean, standard deviation and shape.

The Probability Model tools also apply to finite-state model types, which correspond to models of Markov sources, or equivalently, probabilistic functions of Markov chains. The underlying assumption in using finite-state models is that the system is described at any time as being in one of a finite set of distinct states. The states may correspond to the different program topics of a news programs, that is, local news, weather,

sports, commercials and so forth. At discrete times, the system undergoes a change of state, which is possibly back to the same state, according to a set of probabilities associated with the state. A probability is associated with the likelihood of the transition from sports to weather in news programs. In the special case of discrete-time, first-order Markov chains, the probabilistic dependence between states is truncated such that the probability of the current state depends only on the preceding state. Furthermore, it is assumed that the probabilities are independent of time, which leads to a set of state-transition probabilities.

The Finite State Model tools describe finite-state models, state-transition models and Hidden Markov Models (HMMs). The assumption made for State Transition Models is that each state corresponds to a deterministically observable event, that is, the output of each source being in any given state is not random. Since this model is often too restrictive, the HMM tool includes the case in which the observation is a probabilistic function of the state. The Finite State Model tool defines the abstract base class for state-transition models and discrete HMMs. The finite-state model tool includes an attribute that indicates the number of states, which applies to both state-transition models and discrete HMMs. The state-transition model tool specifies each of the states and gives both the state-transition probabilities as a matrix and the initial state probabilities as a vector. The discrete HMM tool extends the state-transition model tool and provides additional attributes and elements that describe the probabilistic nature of the different observations of each state. The discrete HMM tool includes an attribute that indicates the number of observations per state. It also specifies a sequence that defines the observation symbols and the observation symbol probability distribution of each state [2].

10.3.2 Analytic Models and Cluster Models

This section describes MPEG-7 concepts and tools related to analytic models [1]. The Analytic Model tools describe the association of labels or semantics with collections or classes of multimedia content. The analytic models may be specified by enumerating the items contained in the collection, by describing parameterized representations of the collections in the form of probability models, or both. This allows the analytic models to characterize different semantic concepts by describing the following:

- clusters of multimedia data representing those concepts;
- example collections of multimedia content descriptions; and
- probability models.

The Analytic Model tool is the base type of the tools for describing collection models, probability model classes and cluster models. The Collection Model tool describes the association of semantics with a collection. The Collection Model tool allows the association of a particular semantics with multiple collections. The following describes a collection model consisting of a content collection of three images. The collection of images describes the concept of 'soccer shots on goal'.

```
<CollectionModel confidence="0.75" reliability="0.5" function="described">
        <Label><Name>Soccer shots on goal</Name></Label>
        <Collection xsi:type="ContentCollectionType">
                <Content xsi:type="ImageType">
```

```
                    <Image>
                        <MediaLocator xsi:type="ImageLocatorType">
                            <MediaUri>soccer1.jpg</MediaUri>
                        </MediaLocator>
                    </Image>
                </Content>
                <Content xsi:type="ImageType">
                    <Image>
                        <MediaLocator xsi:type="ImageLocatorType">
                            <MediaUri>soccer2.jpg</MediaUri>
                        </MediaLocator>
                    </Image>
                </Content>
                <Content xsi:type="ImageType">
                    <Image>
                        <MediaLocator xsi:type="ImageLocatorType">
                            <MediaUri>soccer3.jpg</MediaUri>
                        </MediaLocator>
                    </Image>
                </Content>
        </Collection>
</CollectionModel>
```

The Probability Model Class tool describes the association of semantics with classes of descriptions of multimedia content, which are represented using probability models. The probability models provide an aggregate statistical representation of the descriptions in a class. The Probability Model Class tool allows the association of the same semantics with multiple probability models.

10.3.3 Classification Models

This section describes MPEG-7 concepts and tools related to classification models [1]. The Classification Model tool describes information about known multimedia content that can be used to classify unknown multimedia content. The output of the classifiers is the association of labels or semantics with the unknown content. The classifier uses a classification model that describes the association of labels or semantics with known content. The classification model consists of analytic models, which can be content collection models, descriptor collection models or probability class models. In each of these cases, using the group of models, the classifier is able to analyze the unknown content and choose which analytic model best fits the unknown content.

10.4 EXAMPLES OF USAGE

The Cluster Model tool can be used to describe clusters of images. Consider the relevance feedback search scenario illustrated in Figure 10.5. Initially, the user issues a query to an image database. The image database then retrieves images matching the query and returns them to the user. As shown on the left-hand side in Figure 10.5, the user marks which images are relevant to the user's information need, call these positive examples (+), and the remainder are considered as not relevant. This information is then used by

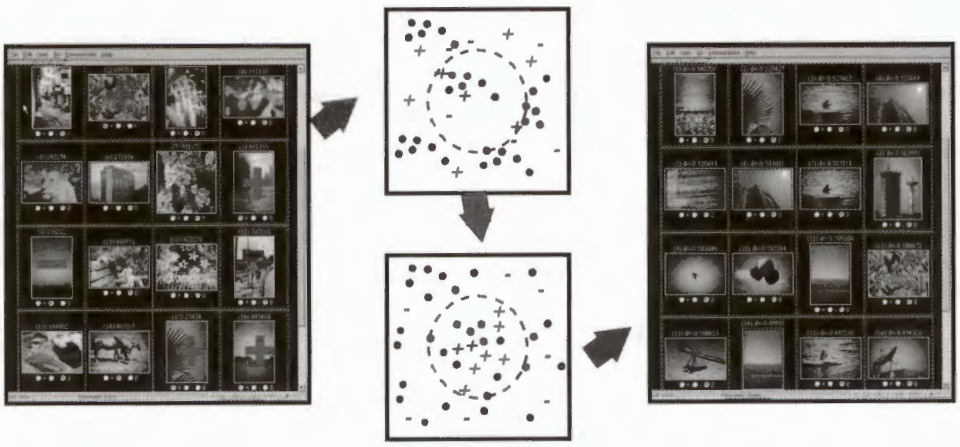


Figure 10.5 Example relevance feedback search scenario for image databases. The clusters of positive and negative examples of images are described using Cluster Model tool

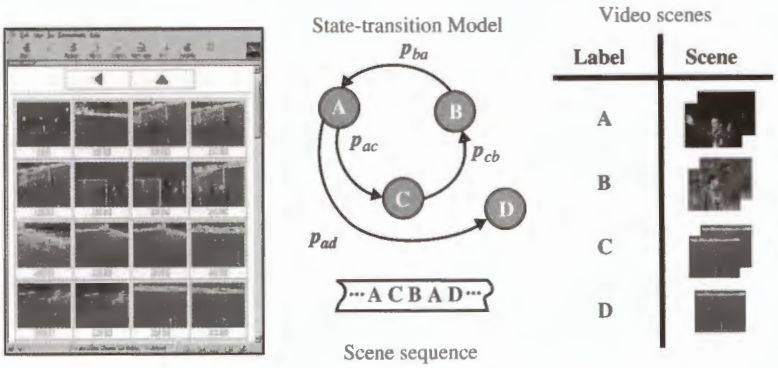


Figure 10.6 Example modeling of soccer video sequence using State Transition Model tool

the retrieval system to transform the metric space that measures similarity of images such that relevant images are clustered together and are returned to the user, as shown on the right-hand side of Figure 10.5 [3]. In this example, the description of the two clusters of images – positive and negative examples, respectively – are described using the Cluster Model tool.

The state-transition model can be used to summarize the transition of events along the temporal dimension of a video sequence. Consider the example in Figure 10.6, which involves a video of a soccer game. In the game, certain types of scenes, such as 'shots on goal', 'goal scores', 'great passes', and 'corner kicks', recur throughout. Each of these scene types can be described using the Model State tool (Analytic Model) that associates semantics with the content. Then, the State Transition Model tool groups together the recurring states and describes the probability of transition between the scenes, as shown in the center and right-hand side in Figure 10.6. This allows the State Transition Model to characterize the entire soccer game video. The State-Transition Model description can

then be compared with other descriptions in order to retrieve similar video sequences or classify the video sequence [4].

REFERENCES

[1] ISO/IEC 15938 (2001), Information Technology – Multimedia Content Description Interface, Singapore.

[2] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, NJ, 1993.

[3] C.-S. Li, J. R. Smith and V. Castelli, "S-STIR: similarity search through iterative refinement," *IS&T/SPIE Symposium on Electronic Imaging: Science and Technology – Storage & Retrieval for Image and Video Databases VI*, San Jose, Calif., 1998, pp. 250–258.

[4] J. R. Smith and C.-S. Li, "Image classification and querying using composite region templates," *Journal of Computer Vision and Pattern Recognition*, **75**(1/2), 165–174 (1999).

11

User Interaction

Peter van Beek[1], Kyoungro Yoon[2] and A. Mufit Ferman[3]

[1]Sharp Laboratories of America, Camas, Washington, USA, [2]LG Electronics, Seoul, Korea, [3]Sharp Laboratories of America, Camas, Washington, USA

11.1 INTRODUCTION

This chapter introduces MPEG-7 tools supporting improved user interaction with multimedia content: a description scheme representing users' preferences and a description scheme representing users' history of content usage. These description schemes can be used to describe the personal preferences and usage patterns of users. Descriptions of users' preferences enable automatic discovery, selection and recommendation or recording of multimedia content. Preferences can be automatically inferred from the user's prior viewing and listening habits, which can in turn be derived from a usage history. Therefore, the main benefit of the tools described in this chapter is improved usability of multimedia devices through personalization: personalized multimedia services offered to the consumer, personalized multimedia content discovery, filtering and selection and personalized consumption of multimedia content. Such tools and services will help reduce the information overload that users may be faced with in the near future, because of the increasing availability of multimedia content from, for example, digital TV broadcast channels and the Internet [1, 2].

Figure 11.1 provides a diagram of the important concepts discussed in this chapter. A user interacts with multimedia content by using a local multimedia system. The multimedia system is the user terminal that is used to find multimedia content, for example, by searching or filtering, and to consume multimedia content, for example, by browsing and viewing or listening. Descriptions of the multimedia content are provided to the system to enable such functionalities. A description of the user's preferences is also provided to the system to enable automatic searching and filtering and personalized browsing of multimedia content. The multimedia system may also generate a usage history description on the basis of a history of the user's interactions with the multimedia content. The usage history descriptions may be used directly for personalized searching, filtering and browsing or may be mapped to a description of the user's preferences by a profiling engine. Both

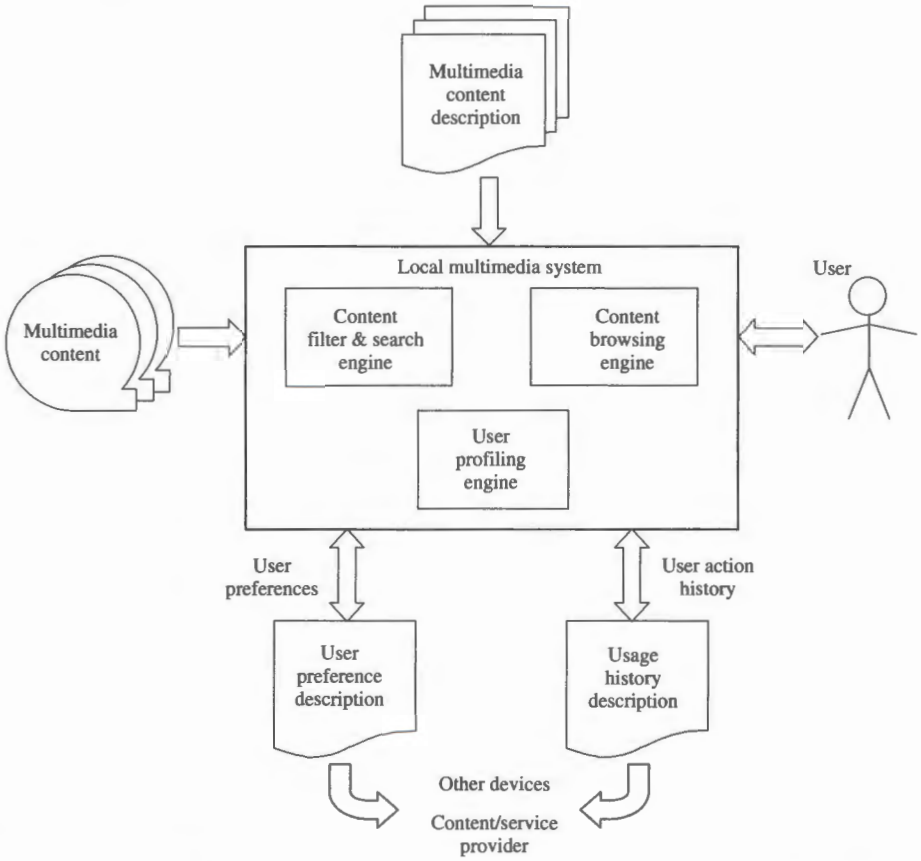


Figure 11.1 Multimedia system and user interaction: multimedia content, multimedia content descriptions, user preference description and usage history description

user preferences descriptions and usage history descriptions may be exchanged with third parties (e.g. service providers) or with other devices. The standardized tools specified by MPEG-7 will support exchange of descriptions of users' preferences and their usage history [3]. There are many benefits to various parties from standardizing an exchange format for user data [4–6]. Multimedia content *consumers* will be able to capture their content preferences, likes and dislikes and enjoy personalized television and music programming. Users may employ software agents to automatically figure out their personal tastes and automatically discover, select and recommend new multimedia content. A standardized format enables users to enter or update their preferences using one device, and then import them into multiple other devices for instant customization. Users can carry a representation of their preferences in a secure smart card or other type of removable storage. Furthermore, *content providers* may find new channels, such as via the Internet, to distribute content targeted at specific customers. *Service providers* can introduce innovative content recommendation services that will enable them to attract and retain customers, by providing the right content. They will be able to develop new business models that derive revenues from sources other than advertising. *Device manufactures*

and *middleware vendors* may find new ways to differentiate their products, by providing improved content filtering and recommendation engines.

It should be noted that the tools for exchanging user profile data introduced in this chapter can be employed in an environment that protects the privacy of end users. Mechanisms to authorize people, organizations or software agents (e.g. by signatures) and to restrict access to user preference information (e.g. by encryption) are not specified in Part 5 of MPEG-7 [7]. However, other standards and technologies in the areas of encryption and authorization are being developed, and can be utilized to provide the necessary information protection [8]. MPEG itself is addressing these issues in its upcoming MPEG-21 standard [9]. Moreover, the tools described here enable individual users to keep a user-owned representation of their profile, stored locally on a client device (rather than having this data stored on remote servers), giving users control over the use of such data. This model enables users to remain anonymous if they choose to, yet allows them to still benefit from various customization capabilities offered by these tools. Individual users can decide on the amount of profile data, if any, that they allow to be generated and wish to share with content or service providers. Users that choose to share (some of) their profile data may enter into contracts with service providers, who will be subject to rules and regulations regarding the use of such data.

In the following section, we introduce the MPEG-7 Usage History description scheme. Then, we introduce the MPEG-7 User Preferences description scheme. Finally, we discuss examples of the usage of these tools and present automatic algorithms for mapping usage history to user preferences, and for content filtering based on user preferences.

11.2 USAGE HISTORY

This section presents a description scheme for describing usage history information gathered over extended periods of time. The collected usage history provides a list of the actions carried out by the user during an observation period, and can subsequently be used by automatic analysis methods to generate user preferences.

Collection and representation of usage history information in a standardized format are relevant to various application areas, and enable numerous usage scenarios. The following list presents some of the functionalities supported by MPEG-7 usage history tools.

- Usage history descriptions can be exchanged directly between consumers, their agents, content providers and devices, under consumer control.
- Users' interactions with a variety of multimedia content types, such as audio and video data, Web pages or content that is locally available to the consumer on his or her electronic equipment (e.g. a DVD player or Personal Video Recorder (PVR)) can be captured and represented.
- Collection and exchange of usage history information is left to the discretion of the consumer, who determines whether his or her consumption history can be collected or distributed, or identity be revealed to third parties.
- A user's content consumption history can be analyzed locally on the client side or remotely at the server location and mapped to a set of preferences, allowing dynamic construction of personalized program or content guides based on user viewing habits.

- Usage history information collected from consumers of multimedia content enables service providers to accurately determine users' responses to specific services and programs. This information can subsequently be employed for developing more efficient content delivery methods, and for targeted delivery of content to individual consumers based on their preferences and demographics.
- Users can be compensated, financially or through value-added services, for sharing their usage history information with advertisers and content providers. Additionally, advanced content access and billing models that utilize the detailed usage history descriptions to offer enhanced 'pay-per-view' services can be introduced.

The context diagram in Figure 11.1 illustrates the way usage history descriptions fit into the general multimedia system. The multimedia system has access to rich multimedia content descriptions, which may be available directly from broadcasters, electronic program guide providers or other third-party services. These descriptions contain metadata associated with the consumable media. A user's actions are monitored and linked to these content descriptions by a usage history collection module, which possesses the ability to monitor the actions that the user performs on a variety of devices, such as audio or video equipment, computer terminals, set-top boxes, personal digital recorders and the like. The monitored actions can correspond to changing the audio or video settings of a device; recording a program; following a hyperlink and so on. The generated usage history information can then be utilized by a local User Profiling Engine to construct User Preference Descriptions, which may in turn be used by a Filtering and Search Engine to assist the viewer in making program selections. Alternatively, the usage history information can be exchanged with a service provider, or be transferred to other devices or equipment, where it can be processed in various ways. Since usage history is expressed in a standardized format, exchange of this information is expedited and carried out without the need for data translation or modification.

Figure 11.2 depicts a diagram of the UsageHistory description scheme in MPEG-7, using notation based on the Unified Modeling Language (UML) [10]. Each rectangular

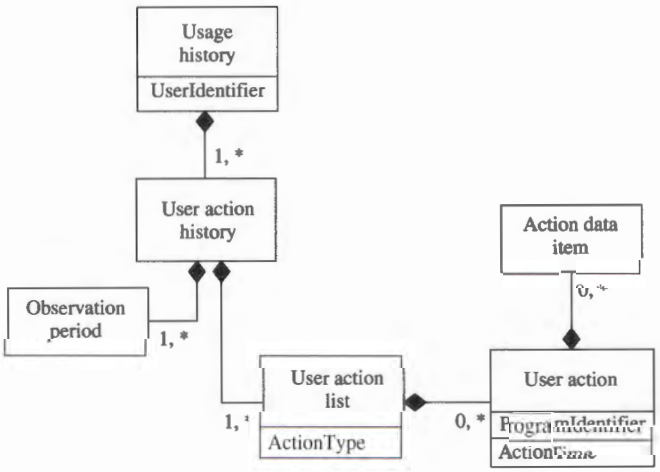


Figure 11.2 UsageHistory description scheme

box corresponds to a description scheme or descriptor; some of these also contain indications of important attributes. Paths between boxes denote composition relationships, and strings such as '1,*' indicate the multiplicity of the relationship. Each UsageHistory description instance is linked, through a UserIdentifier element, to the user or group of users whose content consumption information it provides. The usage history is specified by the UserActionHistory description scheme, which contains multiple lists of actions performed by the user over one or more, nonoverlapping observation periods. Each action list (UserActionList description scheme) is action-type specific; that is, a single list contains actions of a certain type (such as 'PlayStream' or 'Record') only. The types of actions that are recognized and tracked (i.e. the allowed values for the ActionType element) are defined as members of a dictionary; this dictionary can be augmented with new action types to meet the requirements of future applications. Associated with every user action (UserAction description scheme) are the time and duration of the action (ActionTime), an identifier of the multimedia content for which the action took place (ProgramIdentifier) and optional referencing elements that point to related links or resources about the action (ActionDataItem). The time information associated with each user action can be specified in terms of the general time, which denotes the time of occurrence in Coordinated Universal Time (UTC); media time, which denotes the time information that is encoded with a piece of multimedia content, or both. This functionality allows the time information that is associated with such actions as 'FastForward' or 'Rewind' to be provided accurately. It is assumed that descriptions for the content cited in the action history are readily accessible through the provided reference IDs. This approach eliminates the need to keep local copies, and hence duplication, of content descriptions.

The following example provides a brief usage history description instance for user 'John Doe,' who requests that his identity not be revealed to third parties when the description is shared. The collected actions are of type 'Record,' and have been collected over a period of six hours on the evening of October 10, 2001.

```
<UsageHistory allowCollection="true">
   <UserIdentifier protected="true">
      <Name xml:lang="en">John Doe</Name>
   </UserIdentifier>
   <UserActionHistory id="uah-001">
      <ObservationPeriod>
         <TimePoint>2001-10-10T18:00-08:00</TimePoint>
         <Duration>PT6H</Duration>
      </ObservationPeriod>
      <UserActionList id="ual-001" numOfInstances="3"
         totalDuration="PT3H20M">
         <ActionType href="urn:mpeg:mpeg7:cs:ActionTypeCS:2001:1.3">
            <Name>Record</Name>
         </ActionType>
         <UserAction>
            <ActionTime>
               <GeneralTime>
                  <TimePoint>2000-10-09T21:55:00-08:00</TimePoint>
                  <Duration>PT1H10M</Duration>
               </GeneralTime>
            </ActionTime>
            <ProgramIdentifier organization="IDOrg" type="IDName">
               04-flw-000048
```

```
            </ProgramIdentifier>
          </UserAction>
          <!-- More UserAction elements -->
        </UserActionList>
      </UserActionHistory>
    </UsageHistory>
```

11.3 USER PREFERENCES

This section introduces a description scheme that facilitates description of users' preferences pertaining to consumption of multimedia material. The following are some of the functionalities that are supported by the MPEG-7 user preferences tool.

- Automatically discovering, filtering and recommending content by using intelligent software agent technologies. Such agents may adapt to the consumer on the basis of viewing habits and feedback from the user. If allowed, agents may take actions on behalf of the user, for example, to automatically record broadcast content that fits the user's preferences.
- The user preferences description may reside locally and be used as a 'passive' filter on incoming broadcast content, for example, to construct a personalized Electronic Program Guide; on the other hand, in other applications the descriptions may be sent to multimedia servers to 'actively' search, discover and recommend new content.
- Supporting multiple users. Each, logically different, user may be a single physical person or a group of persons. Furthermore, a single person may keep multiple sets of user preferences, for use in different situations. For example, all members of a household can use a single set of user preferences, or, individual members can use separate sets (e.g. to suppress content unsuitable for children).
- Expressing a rich variety of content preferences, such as show title, genre, actor or director and many others, as well as preferences for audiovisual or textual summaries of certain types and durations.
- User preference descriptions can be exchanged between different devices, regardless of brand or physical location. Users can carry User preference descriptions on secure portable media and control the customization of other devices.
- Supporting privacy and user control. Users can indicate whether their preferences (or selected parts) may be accessed by external parties and whether their preference data may be updated automatically or not.

We refer back to the generic diagram in Figure 11.1 to discuss some of the important concepts related to user preference descriptions. A content filter and search engine is part of the multimedia system, as is a content browsing engine. The filter and search engine automatically finds and filters content items that fit the user's personal preferences, by matching the user preference description to descriptions of the multimedia content items. In practice, the output may include identifiers or locators of preferred content. For example, a particular user may indicate a preference for movies of a certain genre, in which case movies of other genres may be filtered out by the filter engine or ranked lower in an ordered list. The content browsing engine may employ the user preference description to tailor the content further, for example, by selecting or presenting to the

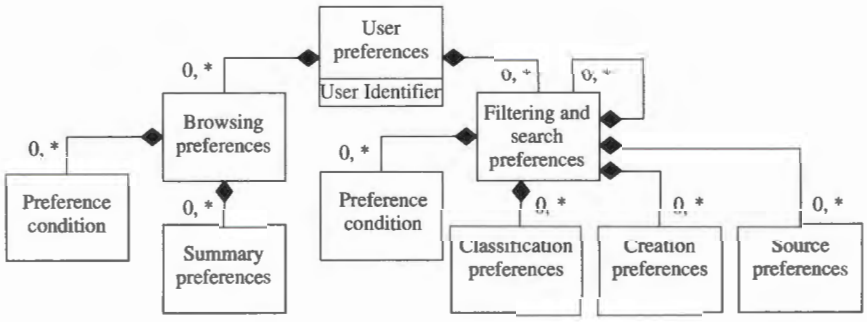


Figure 11.3 UserPreferences description scheme

user multimedia summaries of a certain duration and/or a certain media type. An example is a user who prefers to view only the goals of a soccer match, while another user may prefer a 30-minute highlight summary of the entire match. Both of these are audiovisual summaries that may be selected from a set received from a server or even constructed locally by the browsing engine.

Note that in MPEG-7, the syntax of various descriptors in the user preference description scheme reuses the syntax of the corresponding descriptors in the content description scheme. This guarantees a high level of interoperability between content description aggregators and providers on the one hand and content search and retrieval clients on the other hand.

Figure 11.3 depicts a diagram of the UserPreferences description scheme. The UserPreferences description scheme is associated with a particular user (or group of users) by means of the UserIdentifier descriptor. The UserPreferences description scheme contains two main components, the BrowsingPreferences description scheme and the FilteringAndSearchPreferences description scheme. The FilteringAndSearchPreferences description scheme can be used to specify preferences on the type of content to be searched, filtered, selected and consumed. This description scheme contains the ClassificationPreferences description scheme, CreationPreferences description scheme and SourcePreferences description scheme. To specify hierarchical relations between preferences, a FilteringAndSearchPreferences description may contain child FilteringAndSearchPreferences elements. This can be used to represent specialization of preferences along the hierarchy from most general (at the root) to the most specific (at the leaves).

The CreationPreferences description scheme is used to specify users' preferences related to the following aspects:

- *Title of the content*: For example, title of a favorite show or series;
- *Content creators*: For example, favorite actor, actress, director, presenter, composer, music band;
- *Keywords*: For example, favorite news or documentary topics;
- *Location*: Where the content was created, for example, movies filmed or recorded on location;
- *Period*: When the content was created, for example, during the fifties or seventies;
- *Tool*: What tool or instrument was used to create the content.

The ClassificationPreferences description scheme is used to specify users' preferences related to the following aspects:

- *Country of origin*: Where the content was released or produced;
- *Date*: When the content was first released, for example, first release of a movie;
- *Languages*: For example, the spoken language in radio or TV programs;
- *Form*: Production format, for example, daily news or documentary;
- *Genre*: Favorite genre, for example, science fiction or western;
- *Subject*: Favorite subject;
- *Review*: For example, content that has been rated high by a particular reviewer;
- *Parental guidance*: For example, content with a particular parental rating.

The SourcePreferences description scheme is used to specify users' preferences related to the following aspects:

- *Type*: Dissemination medium or delivery mechanism of content, for example, terrestrial broadcast, web-cast, DVD-ROM;
- *Source*: For example, broadcast TV channel ('BBC-1') or web server;
- *Place*: Location from where content is physically distributed, for example, video rental store;
- *Date*: Time and date of the content availability, for example, time and date of the broadcast, or time and date of availability for on-demand content;
- *Disseminator*: Person or organization making the content available, for example, a publisher, broadcaster or distributor;
- *Media format*: Format of the media, for example, audio- or video-coding format, aspect ratio and so on.

The BrowsingPreferences description scheme can be used to specify preferences regarding browsing and navigation, and in particular contains the following preferences related to summarization:

- *Summary type*: Visual (including e.g. video clips, key frames), audio (including e.g. audio clips, key sounds) or textual;
- *Summary theme*: Favorite topic, theme or type of event;
- *Duration*: Nominal, minimum and maximum duration of audiovisual summaries;
- *Number of key frames*: Nominal, minimum and maximum number;
- *Number of text characters*: Nominal, minimum and maximum number of characters in a textual summary.

The UserPreferences description scheme enables users to specify preferences that apply only in a particular context, in terms of time and place, using the PreferenceCondition descriptor. This descriptor can be associated with the member BrowsingPreferences description scheme and FilteringAndSearchPreferences description scheme in a UserPreference description. The UserPreferences description scheme allows users to specify the relative importance of their preferences with respect to each other, and to express negative preferences and dislikes. To this end, many of the descriptors listed above have an associated numerical '*preference-value*' attribute. The UserPreferences description scheme also enables users to indicate whether their preferences or parts of their preferences should

be kept private or not. Finally, the description scheme enables users to indicate whether the automatic update of their usage preferences description, for example, by an intelligent software agent, should be permitted or not.

The following example provides a brief user preferences description instance for user 'John Doe.' As explained above, the FilteringAndSearchPreferences description scheme in the UserPreferences description scheme is a tree-structured combination of individual preference components. Examples of individual components are the Role, GivenName, FamilyName, DatePeriod elements and the Genre elements in the partial user preference description shown below. Individual components may also contain a preference value, for example, the Genre elements below.

```
<UserPreferences>
  <UserIdentifier protected="true">
    <Name xml:lang="en">John Doe</Name>
  </UserIdentifier>
  <FilteringAndSearchPreferences>
    <CreationPreferences>
      <Creator>
        <Role><Name>Actor</Name></Role>
        <Agent xsi:type="PersonType">
          <Name>
            <GivenName>Harrison</GivenName>
            <FamilyName>Ford</FamilyName>
          </Name>
        </Agent>
      </Creator>
      <DatePeriod>
        <TimePoint>1995-01-01</TimePoint>
        <Duration>P1825D</Duration>
      </DatePeriod>
    </CreationPreferences>
    <ClassificationPreferences>
      <Genre preferenceValue="10"><Name>News</Name></Genre>
      <Genre preferenceValue="20"><Name>Sports</Name></Genre>
    </ClassificationPreferences>
  </FilteringAndSearchPreferences>
</UserPreferences>
```

11.4 EXAMPLES OF USAGE

In the following text, we present a brief overview of methods for mapping usage history data to user preferences and for content filtering based on user preferences. These algorithms are not within the scope of the MPEG-7 standard itself; however, they illustrate the usage of the standard in the context of possible applications.

11.4.1 Mapping Usage History to User Preferences

In this section, we provide a brief example of how user preferences can be extracted from collected usage history information. Mapping usage history records to preference descriptions may in general be viewed as a constrained data mining and knowledge

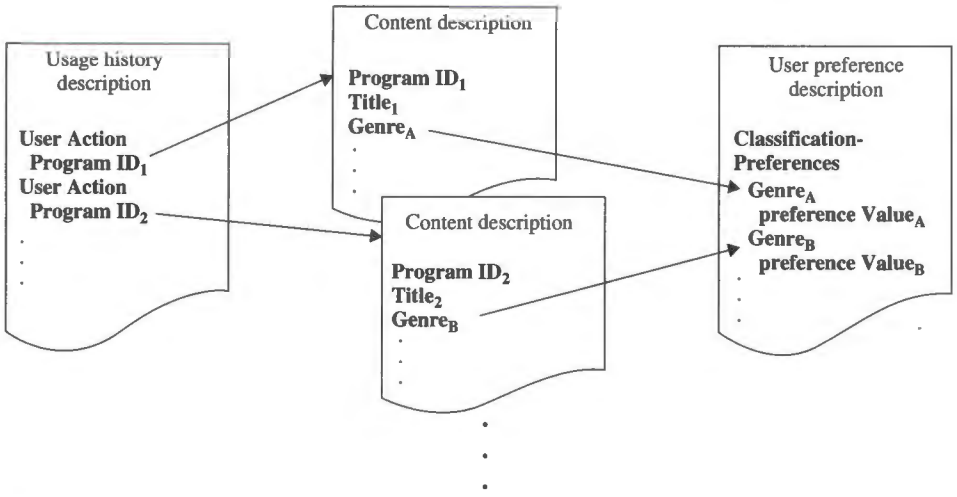


Figure 11.4 Mapping usage history to user preferences: user actions on the left; content descriptions in the middle; user preferences on the right

discovery problem, and can as such be addressed using a variety of available analysis techniques, which include crisp and fuzzy clustering, Bayesian methods, neural networks, rule-based methods, genetic algorithms and belief networks, among others [11, 12].

Figure 11.4 illustrates the typical sequence of events for mapping usage history information to user preferences. Firstly, the content descriptions of the multimedia data that is referenced in the usage history record are collected and analyzed. Secondly, components of the user preference description (e.g. the user's favorite titles, genres, actors etc.) are extracted from these content descriptions and added to the user's profile. Preference values for these components are then computed on the basis of their occurrence frequencies in the history. Additional information about the actions in the usage history can be utilized to generate more accurate user profiles. For example, each action type can be assigned an *a priori* weight that determines how it influences the preference values of content it is associated with. Actions such as Record, SlowMotion, Repeat and PlayStream signal a high level of interest in the content by the consumer, and should therefore have positive weights, whereas other actions like SkipForward, FastForward and SkipToEnd should contribute in a negative way.

Preference values can be computed in batch mode, in which the entire usage history record is considered, or incrementally, using only the most recent history information to update previously calculated values. Batch methods often yield more thorough and detailed preference descriptions, since they take into account a much larger data set; however, they require significantly more storage and computational resources. Incremental algorithms, on the other hand, utilize the recent usage records for each new update, and can therefore modify user preferences and preference values quickly to reflect changes in consumption habits. Generally, incremental methods are more appropriate for local processing of usage history records on the client side.

For example, the preference value y for a particular item in the preference description may be computed using the sigmoid function [13]

$$y = 100\left(\frac{1 - e^{-ax}}{1 + e^{-ax}}\right), \tag{1}$$

where x denotes the normalized total number of occurrences weighted by a coefficient for each action type and a is the slope of the sigmoid function. Preference values are assumed to lie in the range $[-100, 100]$. For incremental updates, Δx, the normalized number of occurrences, should be used instead of x. The updated preference value, y', is then given by

$$y' = 100\left(1 - \frac{1 - e^{(v - a\Delta x)}}{1 + e^{(v - a\Delta x)}}\right), \tag{2}$$

where $v = (100 - y)/(100 + y)$ and y denotes the original preference value.

11.4.2 Filtering Content Using User Preferences

In the following text, we describe a method for automatic filtering of multimedia content using content descriptions and user preference descriptions, such as the example presented in Section 11.3. Each individual component of the FilteringAndSearchPreferences description scheme corresponds to a test or matching operation against components of descriptions of multimedia content. For example, the name values of each Genre element in the sample User preference description in Section 11.3 can be matched against the name values of the Genre element in the following partial content description. Likewise, the date values of the DatePeriod element can be matched against the date value of the Release element of the example content description below.

```
<MultimediaContent xsi:type="AudioVisual">
  <AudioVisual>
    <CreationInformation>
      <Creation>
        <Title xml:lang="en">Afternoon news</Title>
      </Creation>
      <Classification>
        <Genre><Name xml:lang="en">News</Name></Genre>
        <Release date="1998-06-16T21:30+01:00">
          <Region>uk</Region>
        </Release>
      </Classification>
    </CreationInformation>
  </AudioVisual>
</MultimediaContent>
```

The method for testing individual components depends on the nature of the component and may involve (sub)string matching, boolean value matching, numerical value range tests and other operations. Furthermore, each test may result in an exact or a partial match. For example, textual Genre name values are tested using case-insensitive

Table 11.1 Summary of the MPEG-7 tools introduced in this chapter

Tool	Use of tool
Usage history	Collect a history of user actions with respect to multimedia content
User Preferences	Represent content filtering, searching and browsing preferences of a user

string matching. Matching date values, on the other hand, requires numerical comparison operations. The filter algorithm walks through the hierarchy of individual preference components; the individual tests are evaluated and the results are weighted by the preference values.

After individual preference components are tested, the results of these tests must be combined into a single test result that reflects the user's overall preferences. The individual test results can be combined according to the hierarchy of the UserPreferences description. Each parent test becomes a combination of its children tests, and this continues up the hierarchy, yielding one composite test result. The combination of children tests within a single parent may depend on the types of the elements being tested, and may involve boolean or arithmetic operators such as intersection, union, maximum, minimum, averages, and so on. For example, the example user preferences description presented in Section 11.3 specifies a preference for content with an actor having first name 'Harrison' and having last name 'Ford'. Therefore, in this case, an intersection operator can be used to combine the individual test results. As another example, the user preferences description presented above specifies preferences for multiple genres: 'News' and 'Sports'. Different multimedia programs, each with a different genre, may each satisfy the individual tests against these genre preferences. Therefore, in this case, a union operator can be used to combine the individual test results. Generally, entities of the same type are alternatives and should be combined using a union-type operator, while entities of a different type should be combined using an intersection-type operator.

The combination of the user's preferences and the overall matching result against multimedia content descriptions can be used to rank-order the corresponding multimedia content items and/or to automatically filter the content items.

11.5 CONCLUSION

In this chapter, we have introduced MPEG-7 description scheme for effective user interaction with multimedia content. Table 11.1 provides an overview of these tools.

REFERENCES

[1] K. Curtis and O. Draper, "Multimedia content management – provision of validation and personalization services," *Proceedings of International Conference on Multimedia Computing and Systems 1999*, Florence, Italy, 1999, pp. 302–306.

[2] B. Smyth and P. Cotter, "A personalized television listings service," *Communications of the ACM*, **43**(8), 107–111 (2000).

[3] P. Salembier *et al.*, "Description schemes for video programs, users and devices," *Signal Processing: Image Communication*, **16**, 211–234 (2000).
[4] TV Anytime Forum (1999), Call for contributions on: Metadata, Content Referencing and Rights Management, Geneva.
[5] I. Sezan and P. van Beek, "MPEG-7 standard and its expected role in development of new information appliances," *Proceedings of International Conference on Consumer Electronics 2000*, Los Angeles, Calif., 2000, pp. 274–275.
[6] MPEG N4325 (2001), Introduction to MPEG-7, Sydney. See *http://mpeg.cselt.it*.
[7] ISO/IEC 15938, Information Technology – Multimedia Content Description Interface.
[8] M. P. Graven, Leave me alone, PC Magazine, vol. 20, no. 2, January 16, 151–159, 2001.
[9] MPEG N4318 (2001), MPEG-21 Overview, Sydney. See *http://mpeg.cselt.it.*
[10] S. S. Alhir, *UML in a Nutshell*, O'Reilly & Associates, Sebastopol, Calif., 1998.
[11] G. Adomavicius and A. Tuzhilin, "Using data mining methods to build customer profiles," *IEEE Computer*, **34**(2), 74–82 (2001).
[12] K. Cios, W. Pedrycz and R. Swiniarski, *Data Mining Methods for Knowledge Discovery*, Kluwer Academic Publishers, Boston, 1998.
[13] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan College Publishing, New York, 1994.

Section IV
VISUAL DESCRIPTORS

12

Overview of Visual Descriptors

B. S. Manjunath[1] and Thomas Sikora[2]

[1]*Department of Electrical and Computer Engineering, University of California, Santa Barbara, California,* [2]*Heinrich-Hertz-Institute for Communication Technology, Berlin, Germany*

12.1 INTRODUCTION

The main objective of the MPEG-7 visual standard is to provide standardized descriptions of streamed or stored images or video-standardized header bits (visual low-level descriptors) that help users or applications to identify, categorize or filter images or video. These low-level descriptors can be used to compare, filter or browse images or video purely on the basis of non-textual visual descriptions of the content – or in combination with common text-based queries. They will be used differently for different user domains and different application environments.

Selected application examples include digital libraries (image and video catalogues), broadcast media selection (TV channels) and multimedia editing (personalized electronic news service, media authoring). Among this diversity of possible applications the MPEG-7 visual feature descriptors allow users or agents to perform the following tasks taken as examples:

- *Graphics*: Draw a few lines on a screen and get in return a set of images containing similar graphics or logos.
- *Images*: Define objects, including color patches or textures and get in return examples among which you select the ones of interest.
- *Video*: On a given set of video objects, describe object movements, camera motion or relations between objects and get in return a list of videos with similar or dissimilar temporal and spatial relations.
- *Video activity*: On a given video content, describe actions and get a list of videos in which similar actions happen.

The MPEG-7 Visual Descriptors describe basic audiovisual media content on the basis of visual information. For images and video, the content may be described, for

example, by the shape of objects, object size, texture, color, movement of objects and camera motion. It should be noted that MPEG-7 descriptor data may be physically located with the associated AV material, in the same data stream or on the same storage system. Once the MPEG-7 descriptors are available, suitable search engines can be employed to search, filter or browse visual material on the basis of suitable similarity measures. It must be noted that practical search engine implementations may need to match content on the basis of a weighted combination of visual descriptors (for example, color and texture) – and may also include common text-based queries and audio descriptions. The various *Description Schemes* discussed in Chapter 6 facilitate combining descriptors for various applications.

The MPEG-7 descriptors that were developed can be broadly classified into general visual descriptors and domain-specific visual descriptors. The former include color, texture, shape and motion features, while the latter are application-dependent and include a face-recognition descriptor. A brief description of the general visual descriptor classes is given below, and the following chapters in this section describe these generic descriptors in greater detail. The face-recognition descriptor is among the most recent additions to the visual descriptors class and a brief description of the MPEG-7 face-recognition descriptor is presented in Section 12.2.

Color descriptors

Color is one of the most widely used visual features in image and video retrieval. Color features are relatively robust to viewing angle, translation and rotation of the regions of interest. The currently standard six color descriptors represent different aspects of the color feature, including color distribution, spatial layout of color and spatial structure of color.

Texture descriptors

Texture refers to the visual patterns that have properties of homogeneity or not, that result from the presence of multiple colors or intensities in the image. It is a property of virtually any surface, including clouds, trees, bricks, hair and fabric. It contains important structural information of surfaces and their relationship to the surrounding environment. Describing textures in images by appropriate MPEG-7 texture descriptors provides powerful means for similarity matching and retrieval, for both homogenous and nonhomogenous textures.

Shape descriptors

In many image database applications, the shape of image objects provides a powerful visual clue for similarity matching. MPEG-7 provides region and contour descriptors suitable for a variety of applications. Typical examples of such applications include queries on binary images with written characters, trademarks, presegmented object contours and 2-D and 3-D virtual object boundaries. MPEG-7 also defines a 3-D shape descriptor that is useful for invariant (to geometric transformations) recognition of object shapes.

Motion descriptors

All MPEG-7 descriptors described above for color, texture and shape of objects can be readily employed to index images in video sequences. Description of motion features in

video sequences can provide even more powerful clues regarding its content. MPEG-7 has developed descriptors that capture essential motion characteristics into concise and effective descriptions. The most dominant characteristics are provided by camera motion and object motion descriptors.

The four chapters in this section describe in DETAIL the syntax and semantics of the above generic visual feature descriptors. The MPEG-7 face-recognition descriptor is briefly discussed in Section 12.2. During the evaluation phase of the standard, these descriptors were extensively tested. Most of the experiments concerned with similarity retrieval and as such, a suitable similarity measure was needed. The evaluation group finally decided upon a metric that is independent of the data size and normalized to [0, 1]. This metric, referred to as the Average Normalized Modified Retrieval Rate (ANMRR) is described in detail in Section 12.3.

12.2 FACE DESCRIPTOR

Human face perception is an active area in the computer vision community [1]. There are many applications in which automatic face recognition is desirable. Face recognition can be used in image and video retrieval, information security, law enforcement, surveillance and access control. Some examples of face recognition are video database retrieval, user authentication, facility access and advanced video surveillance.

In recent years considerable progress has been made on the problems of face recognition. Different specific techniques were proposed. Among these methods, Principal component analysis (PCA) is a popular technique that is widely used. For initial work on the use of PCA for face recognition, see [2, 3]. The MPEG-7 face descriptor is based on PCA.

The face-recognition descriptor represents the projection of a face vector onto a set of 48 basis vectors that span the space of all possible face vectors. These basis vectors are derived from eigenvectors of a set of training faces and are reasonably robust to view-angle and illumination changes.

12.2.1 Basis Vectors for the Face Images

The basis vectors used in the MPEG-7 are computed as follows. Let Φ_i be a face image in the training set and let $\Gamma_i = \Phi_i - \Psi$ where Ψ is the "average" face image of the training data set. The covariance matrix of the data is thus given by:

$$\mathbf{Q} = \mathbf{A}\mathbf{A}^{\mathrm{T}} \tag{1}$$

where $\mathbf{A} = [\Gamma_0, \Gamma_1, \ldots, \Gamma_{M-1}]$ and $M(M > 48)$ is the number of face images in the training data set. Let $\mathbf{V} = [v_0, v_1, \ldots, v_{47}]^{\mathrm{T}}$ be the eigenvectors of $\mathbf{A}\mathbf{A}^{\mathrm{T}}$ with the largest eigenvalues. Then the eigenfeature vector is defined to be $\tilde{\mathbf{W}} = [\tilde{w}_0, \tilde{w}_1, \ldots, \tilde{w}_{47}]^{\mathrm{T}}$, where $\tilde{w}_i = v_i\Gamma, i = 0, \ldots, 47$.

When we consider a face description, an important characteristic is the symmetry of the faces. Thus, we can consider a flipped face image (flipped along the middle column)

to represent the same person. The features of the original image and the flipped image can both represent the same person. Therefore, one can use the linear combination of the features of the two images to represent the original image. In order to discriminate between the original image and the flipped image, different weights can be used to calculate features.

Suppose Φ be the original image and Φ' be the flipped image of Φ. The face feature is then defined as:

$$w_i = \frac{[\tilde{w}_i(\Phi) + c\tilde{w}_i(\Phi')]}{1+c} \tag{2}$$

where c is the weight given to the eigenfeature vector corresponding to the flipped image.

$$\text{Since } \tilde{w}_i(\Phi') = v_i\Gamma' = v_i'\Gamma, \quad w_i = \frac{1}{1+c}[\tilde{w}_i(\Phi) + c\tilde{w}_i(\Phi')] = \frac{1}{1+c}(v_i\Gamma + cv_i'\Gamma)$$
$$= \frac{1}{1+c}(v_i + cv_i')\Gamma = u_i\Gamma \tag{3}$$

where v_i' is the flipped eigenface of v_i and u_i, $i = 0, \ldots, 47$, are the basis vectors used in the MPEG-7 after normalization.

12.2.2 Face Feature Extraction

A face image should be first normalized before its features can be extracted. This normalized face image is obtained by scaling of the original image to 56 lines with 46 luminance values in each line. The center of the two eyes in each face image should be located on the 24th row and on the 16th and 31st columns.

This normalized image is then used to extract the one-dimensional face vector Φ that consists of the luminance values from the normalized face image arranged into a one-dimensional vector using a column-wise raster scan starting at the top-left corner of the image and ending at the bottom-right corner of the image.

The Face-Recognition feature vector is then calculated by projecting the one-dimensional face vector Φ onto the space defined by the basis matrix $\mathbf{U} = [u_0, u_1, \ldots, u_{47}]^{\mathrm{T}}$ which is specified in Annex 1 of [4]. The features are given by the feature vector $\mathbf{W} = [w_0, w_1, \ldots, w_{47}]^{\mathrm{T}}$ where $\mathbf{W} = \mathbf{U}^{\mathrm{T}}(\Phi - \Psi)$ and Ψ is the mean face vector specified in Annex 2 of [4]. The features are then normalized and quantized as follows:

$$w_i = \begin{cases} -128, & \text{if} \quad w_i/Z < -128 \\ 127, & \text{if} \quad w_i/Z > 127 \\ w_i/Z, & \text{otherwise} \end{cases} \tag{4}$$

where $Z = 16384$ is the normalization constant.

The similarity of face images is simply defined as the Euclidean distance between normalized projections. The eigenvectors with large eigenvalues capture information that

is common to subsets of faces and eigenvectors with small eigenvalues capture information specific to an individual face.

12.3 A QUANTITATIVE EVALUATION OF VISUAL DESCRIPTORS

Experiments were conducted during the MPEG-7 standardization process to compare different competing technologies as well as to optimize adopted methods. Comparing and evaluating technologies for MPEG-7 visual descriptors presented a different set of challenges compared with previous MPEG standardization efforts, since there was no common ground rules for evaluating different methods. For visual descriptors, the retrieval application was found to be the best model to perform experiments. A good retrieval result in response to a visual feature–based query would be a good indicator for the expressiveness of the descriptor. In the experiments, the so-called *query-by-example* (QBE) paradigm has been employed as the primary method for evaluation. In QBE, the respective descriptor values are extracted from the query image, and then matched to the corresponding descriptors of images contained in a database. In order to be objective in the comparisons, a quantitative measure was developed on the basis of the specification of a data set, a query set and the corresponding ground-truth data. The ground-truth data is a set of visually similar images for a given query image.

In defining an objective measure of retrieval effectiveness given a set of queries and the corresponding ground truth, the following factors are considered:

- The measure should be normalized to account for the variation in the size of the ground truth among different queries.
- The measure should favor algorithms that retrieve the ground-truth items as the top matches.
- The measure should assign a penalty for each of the missed ground-truth items. If a ground-truth item is not retrieved within a certain number of top matches, then it is considered as missed.
- The measure should consider the order in which the ground-truth items are retrieved. The measure should favor algorithms that retrieve ground-truth items in highest ranks.
- The measure should assign a penalty for missed ground-truth items. The penalty should be selected such that beyond a certain limit on the rank, it should not matter whether a ground-truth item is found or not, for example, at the 200th or at the 2000th rank.

The following solution was adopted. Consider a query q with a ground-truth size of $NG(q)$; the rank **Rank**(k) of the kth ground-truth image is defined as the position at which this ground-truth image is retrieved (a rank value of one corresponds to the top match). Further, a number $K(q) \geq NG(q)$ is defined that specifies the 'relevant ranks', that is, retrieval with rank larger than $K(q)$ should be considered as a *miss*. For relatively large $NG(q)$ (20–25 items), subjects would judge the retrieval results as useful if items are found within ranks around $2 \times NG(q)$, while for smaller ground-truth sets, even more tolerance would be allowed. For ground-truth items that are not retrieved in the top $K(q)$ ranks, the penalty assigned should be greater than equal to $K(q)$. But a penalty just

equaling $K(q)$ would place retrievals with too many misses at an advantage. A good compromise derived from this reasoning was found by defining a **Rank**(k) as:

$$\mathbf{Rank}(k) = \begin{cases} \mathbf{Rank}(k) & \text{if} \quad \mathbf{Rank}(k) \leq K(q) \\ 1.25 \cdot K(q) & \text{if} \quad \mathbf{Rank}(k) > K(q) \end{cases}$$

$$K(q) = \min\{4 \cdot NG(q),\ 2 \cdot \max[NG(q), \forall q]\} \tag{5}$$

From (A1) we get the Average Rank (AVR) for query q

$$\mathbf{AVR}(q) = \frac{1}{NG(q)} \sum_{k=1}^{NG(q)} \mathbf{Rank} * (k) \tag{6}$$

However, with ground-truth sets of different sizes (actually, NG varies between 3 and 32 in the Common Color Queries (CCQ)), the **AVR** counted from ground-truth sets with small and large $NG(q)$ values would largely differ. To eliminate influences of different $NG(q)$, the *Modified Retrieval Rank*

$$\mathbf{MRR}(q) = \mathbf{AVR}(q) - 0.5 \cdot [1 + NG(q)] \tag{7}$$

is defined, which is always larger than or equal to 0, but with upper margin still dependent on NG. This finally leads to the *Normalized Modified Retrieval Rank (NMRR*)

$$\mathbf{NMRR}(q) = \frac{\mathbf{MRR}(q)}{1.25 \cdot K(q) - 0.5 \cdot [1 + NG(q)]} \tag{8}$$

Note that NMRR(q) can take values between 0 (indicating whole ground truth found) and 1 (indicating nothing found), irrespective of the size of the ground-truth items for query q, $NG(q)$. From (8), it is straightforward to define the *Average Normalized Modified Retrieval Rate* (**ANMRR**), giving just one number indicating the retrieval quality over all queries. This has been used, unless otherwise specified, as the evaluation criterion in all MPEG-7 visual descriptors experiments.

$$\mathbf{ANMRR} = \frac{1}{NQ} \sum_{q=1}^{NQ} \mathbf{NMRR}(q) \tag{9}$$

where NQ is the number of queries. There is evidence that the **ANMRR** measure approximately coincides linearly with the results of subjective evaluation about retrieval accuracy of search engines [5]. It was found in the experiments that there is a strong interrelationship between the compactness of a descriptor (as measured by the numbers of bits needed for the representation) and the retrieval accuracy. This allows the setup of 'rate-accuracy curves' (similar to rate-distortion curves based on signal-to-noise ratio that are widely used in image and video coding).

ACKNOWLEDGMENTS

The authors would like to thank Dr. Lei Wang for contributing to the section on face descriptor, and Prof. Jens Ohm and Dr. Santhana Krishnamachari for their help in writing the section on quantitative evaluation. In addition, our sincere thanks to all the participants in the MPEG-7 video group who contributed to the development of the visual descriptors.

REFERENCES

[1] S. Sirohey, C. Wilson and R. Chellappa, Human and machine recognition of face: A survey," *Proceedings of IEEE*, **83**(5), 705–740 (1995).

[2] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **12**(1), 103–108 (1990).

[3] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, **3**, 72–86 (1991).

[4] ISO/IEC 15938-3:2001, "*Multimedia Content Description Interface – Part 3: Visual*", Version 1, 2001.

[5] P. Ndjiki-Nya *et al.*, "Subjective Evaluation of the MPEG-7 Retrieval Accuracy Measure (ANMRR)", ISO/IEC JTC1/SC29/WG11 (MPEG) document no. M6029, Geneva, May 2000.

13

Color Descriptors

Jens-Rainer Ohm[1], Leszek Cieplinski[2], Heon J. Kim[3], Santhana Krishnamachari[4], B. S. Manjunath[5], Dean S. Messing[6] and Akio Yamada[7]

[1]RWTH Aachen, Institute of Communications Engineering, [2]Mitsubishi Electric ITE-VIL, [3]MI Group, Information Technology Laboratory, LGE, [4]Video Communications, Philips Research, [5]University of California, Santa Barbara, California, [6]Information Systems Technologies Department, Sharp Labs of America, [7]Multimedia Research Laboratories, NEC Corporation

13.1 INTRODUCTION

Color is an important visual attribute for both human vision and computer processing. This chapter provides an overview of MPEG-7 color descriptors. As is the case with the other descriptors, extraction of these descriptors and their use in similarity matching are outside the scope of the normative components of the standard. Nevertheless, efficient extraction and matching techniques are indispensable for a practical system. For each of the descriptors, we provide details of its syntax (and associated semantics, as applicable), descriptor computation (extraction) and experimental results on retrieval effectiveness on a controlled data set with known ground truth. Even though not explicitly stated, all the color descriptors discussed in the following can be computed from arbitrarily shaped image regions as well.

Various factors influenced the selection of these color descriptors. These include

1. Their ability to characterize the perceptual color similarity, judged by performance of the descriptors in matching images and video segments based on color characteristics.
2. Low complexity of the associated extraction and matching techniques, as MPEG-7 systems must be able to handle search and retrieval tasks over large multimedia databases or may be small, portable devices with limited computational power.
3. The sizes of the coded descriptions, which play an important role in indexing and in transmission of the descriptors over bandwidth-limited networks.
4. The scalability and interoperability of the descriptors.

To evaluate the retrieval performance of various color descriptors, experiments based on the *query-by-example* paradigm were conducted. To perform these experiments, a Common Color Dataset (CCD) consisting of about 5000 images and a set of 50 Common Color Queries (CCQ), each with specified ground truth images, was defined [1]. This means that the number of queries was about 1% of the number of images in the database, which guarantees sufficient statistical significance of the results. The CCD consists of a variety of still images, images from stock photo galleries, screen shots of television programs and animations. The query and corresponding ground truth images in CCQ are manually established through a process of visual inspection by different groups of participants. The effectiveness of the individual descriptors is evaluated using the averaged normalized retrieval accuracy measure (ANMRR) (see Chapter 12.3). The final selection of the descriptors was mainly based on their retrieval effectiveness on the CCD as well as their overall complexity.

We now briefly summarize the color descriptors that were defined as part of the standard. The following sections will describe the individual descriptors in more detail.

- The *Color Space Descriptor* allows a selection of a color space to be used in the description. The associated *Color Quantization Descriptor* specifies the partitioning of the given color space into discrete bins. These two descriptors are used in conjunction with other color descriptors.
- The *Dominant Color Descriptor* allows specification of a small number of dominant color values as well as their statistical properties such as distribution and variance. Its purpose is to provide an effective, compact and intuitive representation of colors present in a region or image.
- The *Scalable Color Descriptor* is derived from a color histogram defined in the Hue-Saturation-Value color space with fixed color space quantization. It uses a Haar transform coefficient encoding, allowing scalable representation of description, as well as complexity scalability of feature extraction and matching procedures.
- The *Group of Frames or Group of Pictures Descriptor* is an extension of the Scalable Color Descriptor to a group of frames in a video or a collection of pictures. This descriptor is based on aggregating the color properties of the individual images or video frames.
- The *Color Structure Descriptor* is also based on color histograms, but aims at identifying localized color distributions using a small structuring window. To guarantee interoperability, the Color Structure Descriptor is bound to the HMMD color space (see Section 13.2.2).
- The *Color Layout Descriptor* captures the spatial layout of the representative colors on a grid superimposed on a region or image. Representation is based on coefficients of the Discrete Cosine Transform. This is a very compact descriptor being highly efficient in fast browsing and search applications. It can be applied to still images as well as to video segments.

The main purpose for defining a standardized description is interoperability. This aspect has been extensively investigated during the MPEG-7 experimental evaluation [2, 3],

after which it was concluded to constrain the possible variations in the descriptions. In the context of color descriptors, this has led to specifying a unique choice of color space for each of the color descriptors (with the exception of the dominant color descriptor), as leaving this specification up to the user would have deteriorated the retrieval efficiency and raised complexity of the matching process. As a consequence, the set of histogram-derived descriptors was strictly limited for two variants – Scalable Color and Color Structure – with fixed definition of Color Spaces and limited, interoperable set of bin-quantization choices.

13.2 COLOR SPACES

The Color Space Descriptor specifies a selection of a color space to be used in another color descriptor, specifically, the dominant color descriptor. The color spaces specified in the MPEG-7 are RGB, YCbCr, HSV, HMMD, Monochrome and Linear transformation matrix with reference to RGB. In addition, a flag is provided to indicate reference to a color primary and mapping to a standard reference white value.

The Color Space Descriptor defines the color components as continuous-value entities. For discrete representation, a quantization is necessary. The Color Quantization Descriptor specifies the number of quantization levels for each color component in the color space. A uniform quantization in each of the color components in a given color space is assumed.

The RGB color space is one of the more popular color models. This space is defined as the unit cube in the Cartesian coordinate system. The YCbCr is a legacy color space of the precedent MPEG standards, MPEG-1/2/4. It is defined by a linear transformation of RGB color space as follows:

$$
\begin{aligned}
Y &= 0.299*R + 0.587*G + 0.114*B \\
Cb &= -0.169*R - 0.331*G + 0.500*B \\
Cr &= 0.500*R - 0.419*G - 0.081*B
\end{aligned}
$$

For the Monochrome color representation, Y component alone in the YCrCb is used.

13.2.1 HSV Color Space

The HSV color space defined as a cylinder (see Figure 13.1) consists of Hue, Saturation and Value. Hue (H) represented by the angle from 0 to 360° specifies one color family from another, as red from yellow, green, blue or purple. Saturation (=[0,1]) specifies how pure a color is: pure red, yellow, green, blue and so on. Value (=[0,1]) specifies how bright or dark a color is. The three components are expressed by a nonlinear transform of the three components of RGB color space as shown below:

```
Max = max(R, G, B);
Min = min(R, G, B);
Value = Max;
if( Max == 0 ) then Saturation = 0;
else
```

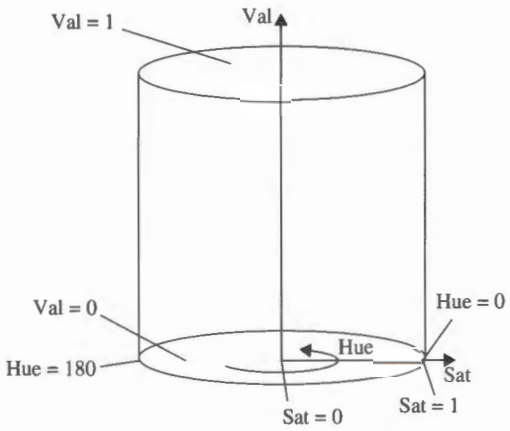


Figure 13.1 HSV color space

```
   Saturation = (Max−Min)/Max;
if( Max == Min ) Hue=0; /* It is achromatic color */
otherwise:
if( Max == R && G >= B )
  Hue = 60*(G−B)/(Max-Min)
else if( Max == R && G < B )
   Hue = 360 + 60*(G−B)/(Max-Min)
else if ( G == Max )
   Hue = 60*(2.0 + (B−R)/(Max-Min))
else
   Hue = 60*(4.0 + (R−G)/(Max-Min))
```

When Max value is equal to Min value (Saturation = 0), it is an achromatic color (white, black or gray). In this case, Hue is set to 0° (means red).

The HSV color space is the color space associated with the scalable color descriptor and the group of frames histogram descriptors (see Sections 13.4 and 13.5). For these two descriptors, the HSV space is uniformly quantized into 256 bins – 16 levels in H, 4 levels in S and 4 levels in V. These two descriptors can also be computed using less than 256 histogram bins. Table 13.1 summarizes the partitioning of the HSV space into 128, 64 and 32 bins and the corresponding number of coefficients used in the Scalable Color and Group of Frames descriptors. See Sections 13.4 and 13.5 for more details.

Table 13.1 Equivalent partitioning of the HSV color space for different numbers of coefficients in the SCD

Number of coefficient ss	Number of bins H	Number of bins S	Number of bins V
16	4	2	2
32	8	2	2
64	8	2	4
128	8	4	4
256	16	4	4

13.2.2 HMMD Color Space

The HMMD (Hue-Max-Min-Diff) color space is closer to a perceptually uniform color space. The double cone shape confines this color space as shown in Figure 13.2. The component names, 'Max', 'Min' and 'Diff' are according to the following transform equations between RGB and HMMD:

```
Max = max(R, G, B);
Min = min(R, G, B);
Diff = Max − Min;
```

Even though the four components are identified in the name of the HMMD color space, one more component, Sum, can be defined.

```
Sum = (Max+Min) /2;
```

Therefore, a total of five components are identified in this color space. However, a set of three components, {H, Max, Min} or {H, Diff, Sum}, is sufficient to form the HMMD color space and specify a color point. The semantics of each component is distinct and described as follows. Hue ($H = [0°, 360°]$) has the same property as Hue in the HSV color space. Max (=[0,1]) specifies how much black color is present, giving the flavor of shade or blackness. Max has the same RGB related transform as Value in HSV but the valid subspace is different in HMMD. Thus, the interpretation is different from Value. Min (=[0,1]) specifies how much white color is present, giving the flavor of tint or whiteness. Diff (=[0,1]) specifies how much a color is close to pure colors, giving the flavor of tone or colorfulness. It has a similar property as Saturation in HSV but the valid subspace is again different. Finally, Sum (=[0,1]) specifies the brightness of the color.

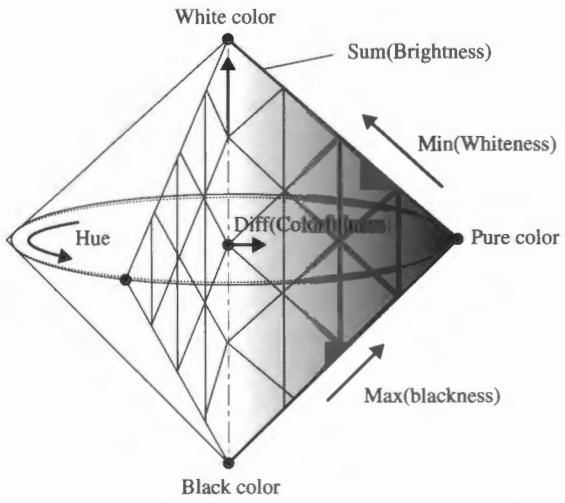


Figure 13.2 (see also Plate 1) Double cone representation of the HMMD color space

HMMD color space quantization

This subsection describes the nonuniform quantization of the HMMD color space used by the color structure descriptor.

As already discussed, the HMMD space can be defined in three dimensions using *sum*- and *diff*-axes as well as *hue angle* as in Figure 13.2. A 3-D quantization of such a space corresponds to a partition[1] of the space into 3-D cells. Four nonuniform quantizations of HMMD are defined in the MPEG-7 Standard. The four quantizations partition the space into 256, 128, 64 and 32 cells, respectively.

Each 3-D quantization is defined via five *subspaces* of HMMD as follows. The *diff*-axis, itself defined on the interval [0, 255], is cut into five subintervals: [0,6), [6, 20), [20, 60), [60, 110) and [110, 255). This 1D partition of the *diff*-axis implicitly defines five subspaces numbered 0, 1, . . . , 4, respectively. Each subspace is that subset of HMMD where *sum* and *hue* are allowed to take all values in their respective ranges and where *diff* is restricted to one of the five intervals.

A partition of HMMD is obtained by partitioning the ranges of *hue* and *sum* into *uniform* intervals within each subspace according to Table 13.2. The table actually consists of four tables (one for each quantization of HMMD), the columns of which are alternately white or shaded. For each quantization, the table tells how to partition the subspaces to yield the overall partition.

For example, to partition HMMD space into 128 cells, the table instructs us to partition Subspace 4 by cutting *hue* into 8 uniform intervals and *sum* into 4 uniform intervals, giving 32 cells in Subspace 4. The other 4 subspaces are partitioned in a similar manner to yield the overall *nonuniform* quantization of HMMD into 128 cells.

We note that for the 32-cell quantization, the dividing line between subspaces 1 and 2 is missing in the table. This indicates that the two subspaces have been united

Table 13.2 HMMD subspace quantization for each of the four partitions of the space

No. of cells	256		128		64		32	
Subspace	Hue	Sum	Hue	Sum	Hue	Sum	Hue	Sum
0	1	32	1	16	1	8	1	8
1	4	8	4	4	4	4	4	4
2	16	4	8	4	4	4		
3	16	4	8	4	8	2	4	1
4	16	4	8	4	8	1	4	1

[1] A *partition* of a space, S, is a collection, $\{p_1, \ldots, p_N\}$, of cells such that $p_i \subseteq S$, $p_i \cap p_j = \emptyset$ for $i \neq j$, and $\bigcup_{i=1}^{N} p_i = S$.

into a single subspace. Thus, the 32-cell partition of HMMD is defined via 4 subspaces instead of 5.

Figure 13.3 depicts a slice of HMMD color space in the *diff-sum* plane for zero hue angle and shows the cells for the 128-cell quantization. Subspace boundaries are indicated in the figure by vertical lines in the plane. The *diff*-axis values that determine the subspace boundaries are shown in black at the top of the dashed cut-point markers along the upper edge of the plane.

Horizontal lines within each subspace depict the division of the *sum*-axis into uniform intervals. The grey rotation arrows around each cut-point marker indicate the partition of the *hue angle*. The grey number to the right of a rotation arrow corresponds to the number of intervals into which the range of *hue* has been partitioned in the subspace to the right of the cut-point. For example, Figure 13.3 states that the range *hue* associated with the subspace between *diff* = 60 and *diff* = 110 (i.e. subspace 3) is divided into 8 equal intervals. This agrees with the entry in Table 13.2.

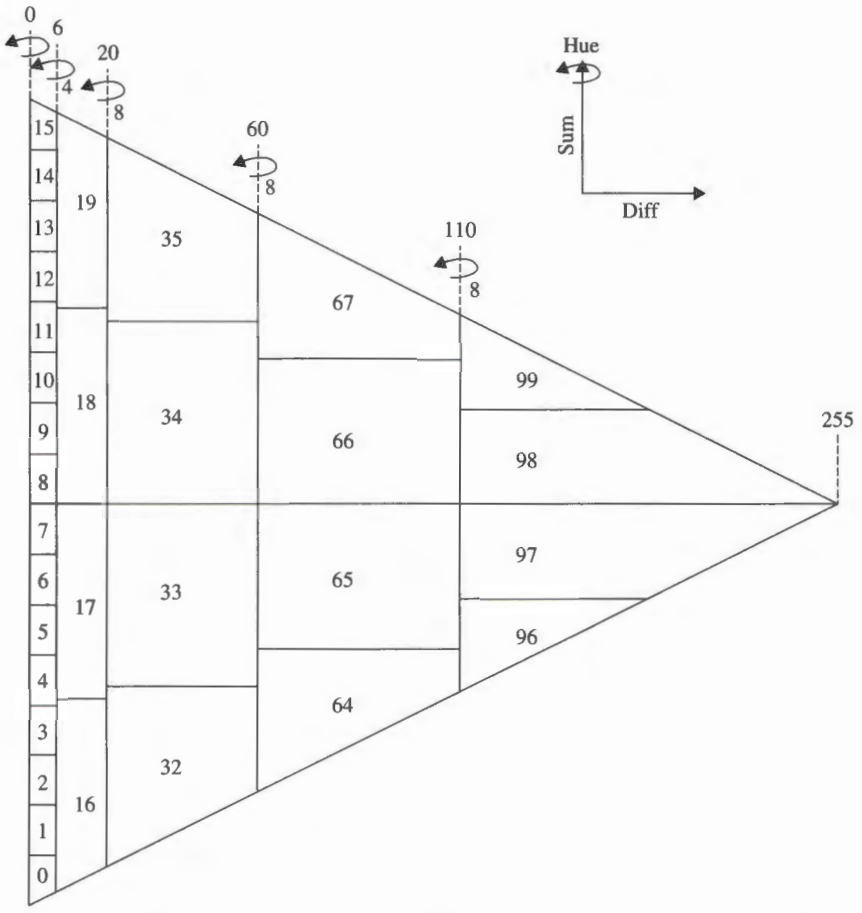


Figure 13.3 A slice of 128-cell quantized HMMD color space at *hue* = 0, and the indexing scheme used to number the cells

Finally, Figure 13.3 indicates the scheme for numbering the cells in a partition of HMMD space. This is important because of the 1-to-1 association between cells and the CSD bin indices discussed in Section 13.6.

Besides the five color spaces described up to now, the Color space descriptor can describe any 3×3 color transform matrix that specifies the linear transformation between RGB and the respective color space. Thus, any linear transformation from the RGB color space can be specified in the MPEG-7 Color space descriptor.

13.3 DOMINANT COLOR DESCRIPTOR

The dominant color descriptor (DCD) provides a compact description of the representative colors in an image or image region. Its main target applications are similarity retrieval in image databases and browsing of image databases based on single or several color values. Unlike the traditional histogram-based descriptors, the representative colors are computed from each image instead of being fixed in the color space, thus allowing the color representation to be accurate and compact. The DCD allows for efficient indexing[2] of large databases as presented in [4].

The DCD is defined to be

$$F = \{(\mathbf{c}_i, p_i, v_i), s\}, \quad (i = 1, 2, \ldots, N)$$

most where N is the number of dominant colors. Each dominant color value $\mathbf{c}_i$ is a vector of corresponding color space component values (e.g. a 3-D vector in the RGB color space). The percentage p_i (normalized to a value between 0 and 1) is the fraction of pixels in the image or image region corresponding to color c_i, and $\sum_i p_i = 1$. The optional color variance v_i describes the variation of the color values of the pixels in a cluster around the corresponding representative color. The spatial coherency s is a single number that represents the overall spatial homogeneity of the dominant colors in the image. The number of dominant colors N can vary from image to image, and a maximum of eight dominant colors was found to be sufficient to represent an image or an image region. The color space quantization depends on the color space specifications defined for the entire database and need not be specified with each descriptor.

The binary syntax of the DCD specifies three bits to represent the number of dominant colors and five bits for each of the percentage values (uniform quantization of [0,1)). The Color Space and Color Quantization descriptors are referred to by this descriptor and RGB is the default color space. The optional color variances are encoded

[2] Indexing in the database context refers to an efficient pruning of the search space so as to minimize the number of distance computations and disk I/O accesses needed for computing the nearest neighbors of a given query feature vector. The feature vector dimensions of typical visual descriptors are quite large. For example, the number of bins in a histogram descriptor may be of the order of few hundreds. It is a well-known fact that nearest-neighbor search for similarity retrieval in such high-dimensional spaces is quite expensive and is often refered to as the *dimensionality curse* in the database literature.

at one bit per color with nonuniform quantization. The table below summarizes the binary syntax of the DCD. See Reference 5 for detailed specifications.

Field	Number of Bits	Meaning
NumberofColors	3	Specifies number of dominant colors
SpatialCoherency	5	Spatial Coherency value
Percentage[]	5	Normalized percentage associated with each dominant color
ColorVariance[] []	1	Color variance of each dominant color
Index[] []	1–12	Dominant color values

13.3.1 Extraction

The extraction procedure for the dominant color described in Reference 6 uses the Generalized Lloyd Algorithm [7] to cluster the pixel color values. It is recommended that the clustering be performed in a perceptually uniform color space such as the CIE LUV. The distortion D_i in the ith cluster is given as

$$D_i = \sum_n h(n)\|\mathbf{x}(n) - c_i\|^2, \quad \mathbf{x}(n) \in C_i$$

where c_i is the centroid of cluster C_i, $\mathbf{x}$ (n) is the color vector at pixel n and $h(n)$ is the perceptual weight for pixel n. The perceptual weights are calculated from local pixel statistics to account for the fact that human visual perception is more sensitive to changes in smooth regions than in textured regions [8]. The update rule for the above distortion metric can be derived to be:

$$c_i = \frac{\sum h(n)\mathbf{x}(n)}{\sum h(n)}, \quad \mathbf{x}(n) \in C_i$$

The procedure is initialized with one cluster consisting of all pixels and one representative color computed as the centroid (center of mass) of the cluster. The algorithm then follows a sequence of centroid calculation and clustering steps until a stopping criterion (minimum distortion or maximum number of iterations) is met. The clusters with highest distortion are divided by adding perturbation vectors to the centroids until the maximum distortion falls below a predefined threshold or the maximum number of clusters is generated. The percentage or fraction of pixels in the image belonging to each of the quantized colors is then calculated and these resulting percentages are uniformly quantized to five bits. The color values are quantized according to the specifications of the color space and the associated color-quantization descriptors.

A simple connected component analysis is performed to identify groups of pixels of the same dominant color that are spatially connected. Four connectivity (the four nearest

neighbors of a pixel) is assumed. The normalized average number of the connecting pixels of each dominant color is then computed. A 3×3 masking window is used for this purpose. This is used as a measure of spatial coherency for that dominant color. The overall spatial coherence is then a linear combination of the individual spatial coherence values with the corresponding percentages p_i being the weights. The spatial coherence value is then nonuniformly quantized to 5 bits, where 31 means highest confidence and 1 means no confidence. The value zero is used for cases in which it is not computed. Finally, the color variances are computed as variances of the pixel values within each cluster and nonuniformly quantized to one bit per color component.

13.3.2 Similarity Matching

Each object or region in the database is represented using the DCD as defined above. Typically, three to four colors provide a good characterization of the region colors. Given a query image, similarity retrieval involves searching the database for similar color distributions as the input query. Since the number of representative colors is small, one can first search the database for each of the representative colors separately, and then combine the results. Searching for individual colors can be done very efficiently in a 3-D color space.

Consider two DCDs,

$$F_1 = \{(\mathbf{c}_{1i}, p_{1i}, v_{1i}), s_1\}, \quad (i = 1, 2, \ldots, N_1) \text{ and}$$
$$F_2 = \{(\mathbf{c}_{2i}, p_{2i}, v_{2i}), s_2\}, \quad (i = 1, 2, \ldots, N_2).$$

Ignoring the optional variance parameter and the spatial coherence, the dissimilarity $D(F_1, F_2)$ between the two descriptors can be computed as:

$$D^2(F_1, F_2) = \sum_{i=1}^{N_1} p_{1i}^2 + \sum_{j=1}^{N_2} p_{2j}^2 - \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} 2a_{1i,2j} p_{1i} p_{2j}$$

where the subscripts 1 and 2 in all variables stand for descriptions F_1 and F_2, respectively, and $a_{k,l}$ is the similarity coefficient between two colors c_k and c_l,

$$a_{k,l} = \begin{cases} 1 - d_{k,l}/d_{\max} & d_{k,l} \leq T_d \\ 0 & d_{k,l} > T_d \end{cases}$$

where $d_{k,l} = \|c_k - c_l\|$ is the Euclidean distance between two colors c_k and c_l, T_d is the maximum distance for two colors to be considered similar and $d_{\max} = \alpha T_d$. In particular, this means that any two dominant colors from one single description are at least T_d distance apart. A recommended value for T_d is between 10 to 20 in the CIE-LUV color space and for α is between 1.0 to 1.5. The above dissimilarity measure can be shown to be equivalent to the quadratic distance measure that is commonly used in comparing two color histogram descriptors.

One variation of the above distance is to use the spatial coherence field. For example, in the MPEG-7 experiments, the following distance was used:

$$D_S = w_1 \text{abs}(s_1 - s_2) D + w_2 D$$

where s_1 and s_2 are the spatial coherencies of the query and target descriptors and w_1 and w_2 are fixed weights, with recommended settings to 0.3 and 0.7, respectively.

This distance can be modified to take into account the optional variance. If the color variance field is present, the matching function is based on modeling of the color distribution as a mixture of Gaussian distributions with parameters defined as color values and color variance [4]. Calculation of the squared difference between the query and target distributions then leads to the following formula for the matching function:

$$D_V = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} p_{1i} p_{1j} f_{1i\ 1j} + \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} p_{2i} p_{2j} f_{2i\ 2j} - \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} 2 p_{1i} p_{2j} f_{1i\ 2j}$$

where

$$f_{x_i\ y_j} = \frac{1}{2\pi \sqrt{v_{x_i\ y_j}^{(l)} v_{x_i\ y_j}^{(u)} v_{x_i\ y_j}^{(v)}}} \times \exp\left[-\left(\frac{c_{x_i\ y_j}^{(l)}}{v_{x_i\ y_j}^{(l)}} + \frac{c_{x_i\ y_j}^{(u)}}{v_{x_i\ y_j}^{(u)}} + \frac{c_{x_i\ y_j}^{(v)}}{v_{x_i\ y_j}^{(v)}} \right) \Big/ 2 \right]$$

and

$$c_{x_i\ y_j}^{(l)} = (c_{x_i}^{(l)} - c_{y_j}^{(l)})^2,\ v_{x_i\ y_j}^{(l)} = (v_{x_i}^{(l)} + v_{y_j}^{(l)})$$

In the equations above, $c_{x_i}^{(l)}$ and $v_{x_i}^{(l)}$ are dominant color values and color variances, x, y index the query and target descriptors, i, j index the descriptor components and l, u and v the components of the color space.

13.3.3 Experimental Results

Table 13.3 shows a comparison of ANMRR results for two average descriptor sizes using the common color datasets and queries, CCD and CCQ. It can be seen that reasonable results are obtained even for the basic version of the descriptor and a significant improvement can be achieved by using the optional fields. Table 13.4 gives results using the spatial variance parameter and comparing with the DC descriptor (without variance). These experiments suggest that using five bits for the spatial coherence field is a reasonable trade-off between complexity and effectiveness of the descriptor. It should be noted that one of the main objectives of the DCD is to provide a compact and intuitive representation of salient colors in a given region of interest.

Table 13.3 ANMRR results for dominant color

Average number of colors	ANMRR(D)	ANMRR(D_S)	ANMRR(D_V)
3	0.31	0.30	0.25
5	0.25	0.21	0.16

Table 13.4 ANMRR results for the dominant color with spatial coherence. An average of 5.3 colors per image are used for the MPEG-7 common color dataset. While assigning the bits to individual dominant colors gave better performance, the increased complexity of the descriptor was the main factor in choosing a single spatial coherence value

Number of bits for the spatial coherence	ANMRR	
	Spatial coherence field with dominant colors	Spatial coherence for each dominant color
5	**0.221**	
4	0.227	
3	0.246	
2	0.250	**0.197**
1	0.252	0.202
0	**0.252** (without spatial coherence value)	

13.4 SCALABLE COLOR DESCRIPTOR

The scalable color descriptor (SCD) can be interpreted as a Haar transform–based encoding scheme applied across values of a color histogram in the HSV color space (see Section 13.2.1). The histogram values are extracted, normalized and nonlinearly mapped into a four-bit integer representation, giving higher significance to small values. The Haar transform is applied to the four-bit integer values across the histogram bins. The basic unit of the transform consists of a sum operation and a difference operation (see Figure 13.4a), which relate to primitive low-pass and high-pass filters. Summing pairs of adjacent bins is equivalent to the calculation of a histogram with half number of bins. From the sums of every two adjacent Hue bin values out of the 256-bin histogram, we get a representation of a 128-bin histogram with 8 levels in H, 4 levels in S and 4 levels in V. If this process is repeated, the resulting 64, 32 or 16 sum coefficients from the Haar representation are equivalent to histograms with 64, 32 or 16 bins. Table 13.1 shows the equivalent partitioning of the HSV color space for different number of coefficients of the Haar transform. If an application does not require the full resolution, limited number of Haar coefficients may simply be extracted from a 128-, 64- or 32-bin histogram; this would still guarantee interoperability with another representation in which all coefficients were extracted, but only to the precision of the coefficients that are available in both of the representations. Note that since all partitions in the original color space quantization are powers of 2, the combination with the Haar transform appears to be very natural.

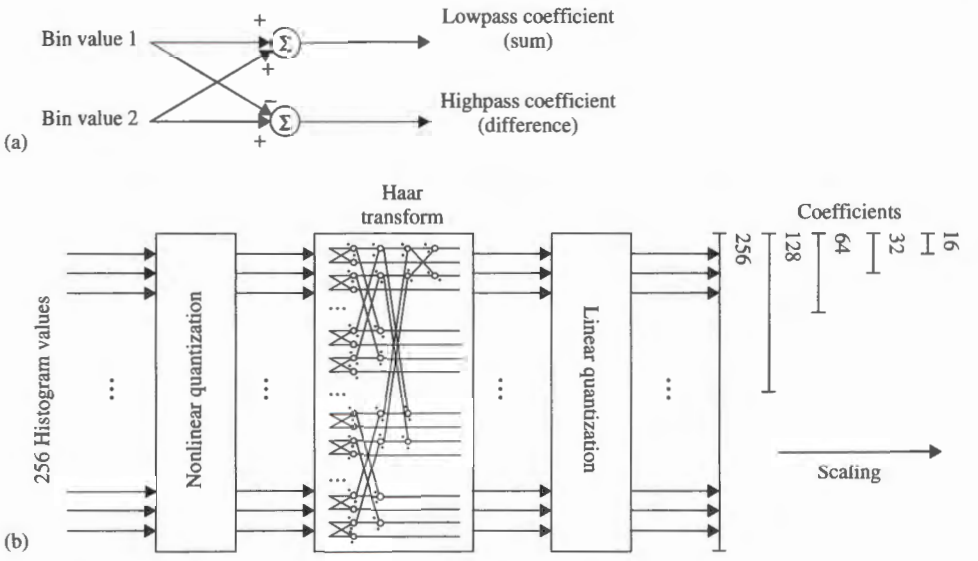


Figure 13.4 (a) Basic unit of Haar transform and (b) a schematic diagram of SCD generation

The high-pass (difference) coefficients of the Haar transform express the information contained in finer-resolution levels (with higher number of bins). Histograms of natural image signals usually exhibit high redundancy between adjacent histogram bins. This can be explained by the 'impurity' (slight variation) of colors caused by variable illumination and shadowing effects. Hence, it can be expected that the high-pass coefficients expressing differences between adjacent histogram bins usually have only small values. Exploiting this property, it is possible to truncate the high-pass coefficients to an integer representation with only a small number of bits.

13.4.1 Extraction and Matching

Figure 13.4(b) shows the block diagram of the SCD extraction process. The output representation is scalable in terms of number of bins, by varying the number of coefficients used. Interoperability between different resolution levels is retained because of the scaling property of the Haar transform. Thus, matching based on the information from subsets of coefficients guarantees an approximation of the similarity in full resolution. Furthermore, as mentioned above, also the feature extraction operation can be scaled to lower levels (less bins in the source histogram).

Besides the scalability in the number of histogram bins, another form of scalability is achieved by scaling the quantized (integer) representation of the coefficients to different number of bits. The 'difference' coefficients in the Haar transform can take either positive or negative values. The sign part is always retained, whereas the magnitude part can be scaled by skipping the least significant bits. Using only the sign bit (one bit/coefficient) leads to an extremely compact representation, while good retrieval efficiency is retained.

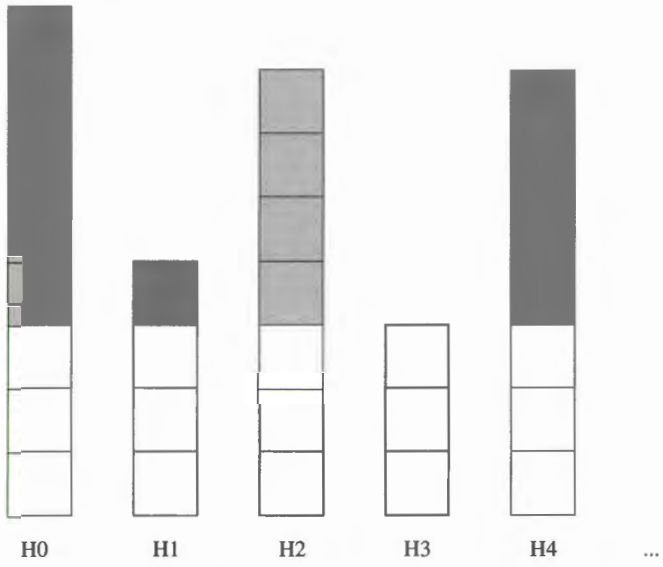


Figure 13.5 Illustration of bit-plane scalability

At the highest accuracy level, one to eight bits are defined for integer representations of the magnitude part, depending on the relevance of the respective coefficients. Between these extremes, it is possible to scale to different resolution levels. For example, consider a set of five coefficients whose magnitudes are encoded using eight, four, seven, three and seven bits, respectively, as shown in Figure 13.5. If the lowest three bits are discarded in the scalable bit representation, only five, one, four, zero and four bits remain to encode the absolute value.

l_1-norm based matching (sum of absolute differences) can be applied in the Haar transform domain; however, results are not identical with l_1-norm based matching in the histogram domain. In the case in which only the sign bit is used (all bit planes representing the absolute value discarded), the l_1-norm degenerates to a Hamming distance, allowing very low complexity in the distance calculation.

13.4.2 Representation

The scalability in the number of histogram bins and the number of bit planes are represented by the fields *NumberofCoefficients* and *NumberofBitplanesDiscarded*. The *NumberofCoefficients* is used to indicate whether 16, 32, 64, 128 or 256 bins (coefficients) are used. The *NumberofBitplanesDiscarded* specifies the number of bit planes of the coefficients that are discarded, ranging from zero to eight. In the case this value is eight, the magnitude of the coefficients are not present, but only the sign of each coefficient is retained, which is represented by the *CoefficientSign*. The magnitudes of the coefficients are represented in a bit-plane fashion, which means that the most significant bits of all coefficients are taken first, followed by the next most significant, and so on. The bit-plane

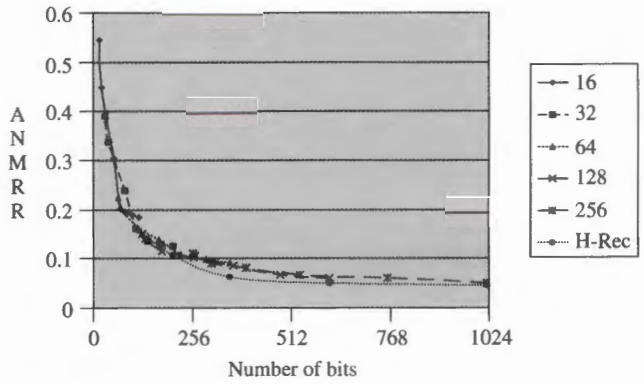


Figure 13.6 (see also Plate 2) Retrieval results with different number of Haar coefficients (16–256) quantized at different number of bits. H-Rec signifies retrieval results after reconstruction of histogram from Haar coefficients at full-bit resolution, which constitutes a lower limit of the efficiency curve

representation allows the transmission of only a certain amount of most significant bits for bandwidth-constrained applications. The representation is as follows:

Field	Number of bits	Meaning
NumberofCoefficients	3	Specifies the number of histogram bins = 16,32,64,128,256
NumberofBitplanesDiscarded	3	Specifies discarding 0 to 8 bit planes
CoefficientSign[]	NumberofCoefficients	The sign of each coefficient
BitPlane[] []	See text	Coefficient magnitudes represented in a bit-plane fashion

13.4.3 Experimental Results

Retrieval results achieved by the SCD are shown in Figure 13.6. In addition, the ANMRR quality measure was calculated from matching in the histogram domain, after performing an inverse Haar transform. The results show that a reasonable performance can be achieved even with small number of bits, while the performance saturates between 256 and 512 bits.

13.5 GROUP-OF-FRAME OR GROUP-OF-PICTURE DESCRIPTOR

The Group-of-frame (GoF) or Group-of-picture (GoP) color descriptor is used for the joint representation of color-based features for multiple images or multiple frames in a video segment. This descriptor can be used to represent a collection of contiguous or

noncontiguous video frames or a group of images. Traditionally for a group of frames or pictures, a key frame or a key image is selected from such a group and the color-related features of the entire collection are represented by those of the chosen sample. Such methods are highly dependent on the quality of the representative sample selection and may lead to unreliable results. The GoF or GoP color descriptors are histogram-based descriptors that reliably capture the color content of multiple images or video frames.

13.5.1 Extraction and Matching

The GoF or GoP color descriptors are obtained by aggregating the histograms of multiple images or video frames and representing the aggregated histograms using the SCD. The individual image or video frame histogram is computed on the basis of the uniform quantization of the HSV color space as detailed in Table 13.1. Three different ways are defined to compute the aggregate color histogram values for the whole series of images or video frames: average, median or intersection aggregation. The aggregated histogram is then input to the Haar transform to build the SCD representation as presented in the previous section.

The average histogram is computed by accumulating the frame or picture histograms in the group and subsequently normalizing each accumulated bin value by N, where N is the number of frames in the GoF or the number of images in the GoP. The average histogram is simple to compute. The descriptor can be updated easily if additional images or video frames are added to the group. A potential problem with using sample averages to compute the GoF or GoP histogram is the sensitivity of the mean operator to outliers. The median histogram is obtained by constructing, for each bin, the ascending list of N frame or picture histogram values over the length of the GoF or GoP and assigning the median of this list to the corresponding bin in the GoF or GoP histogram. The median histogram eliminates aberrant effects such as lighting changes, occlusion, text overlays, and so forth, which the average histogram is vulnerable to. One concern regarding the use of the median histogram is the increased computational complexity.

The intersection histogram (Int_Histogram) is obtained by computing for each bin the minimum value over all the N frame or picture histograms in the group. Each bin value in the intersection histogram thus represents the number of pixels of a particular color that appear in all of the GoF frames. The intersection histogram is characteristically different from the average and median histograms in that it provides the 'least common' color traits of the given GoF or GoP rather than an estimate of the color distribution.

The matching for the GoF or GoP descriptor is performed exactly similar to the SCD descriptor. It should be ensured that the aggregation method used for the descriptors that are being matched are the same.

13.5.2 Descriptor Representation

As alluded to before, the GoF or GoP color descriptor is an extension of the SCD. The representation for the GoF or GoP descriptor is identical to the SCD with an additional attribute *aggregation*. The three different possible methods of aggregation is

represented by this attribute using two bits. This is followed by the associated SCD descriptor.

Field	Number of bits	Meaning
aggregation	2	Specifies the three different types of aggregation
ScalableColorDescriptor	See Section 13.4.2	Specifies the SCD

13.5.3 Experimental Results

The joint representation of a collection of video frames or images obtained using the GoF or GoP color descriptor can be used in different applications in video content management, namely, query-by-example (QBE) based retrieval applications, shot grouping, fast search and browsing of an image or video database. A description of the experimental results for video segment matching is presented here. The experiments were conducted on about three hours of video in the MPEG-7 data set. The data set contained various types of video, including sports programs, news clips, videos of natural scenes and so forth. Shot segmentation was performed on this data using cut, dissolve, fade, and wipe detection resulting in 1544 shots (groups-of-frames). For each GoF, histograms were computed for the individual frames and the three different types of aggregation were performed to obtain the GoF descriptor. In addition, for each GoF, a key frame was identified to compare the performance GoF descriptor–based search against the key frame–based search. The key frame was selected by searching through all the frames in a GoF to find the optimal frame that had the minimum mean absolute error with all the other frames within that GoF. From the 1544 GoFs, 31 queries were selected and for each query a set of ground truth GoFs were manually identified [9]. The queries ranged from almost static GoFs to dynamic scenes with edit effects and transitions. Table 13.5 shows the ANMRR retrieval results using the average and median aggregation–based GoF matching against the key frame–based matching. From the table, it can be seen that the GoF descriptor performs better than the optimum key frame–based matching. Results of another application to find if a given frame belongs to a GoF using the intersection aggregation can be found in [10]. Experimental results on the use of GoP descriptor for fast search of image databases and content-based browsing can be found in [11, 12].

Table 13.5 Comparison of ANMRR video segment retrieval results using average, median GoF descriptor and the key frame–based histogram

	L_1	L_2
GoF-Average	0.041367	0.089982
GoF-Median	0.042614	0.090640
Optimal key frame	0.053500	0.101852

13.6 COLOR STRUCTURE DESCRIPTOR

The color structure descriptor (CSD) represents an image by both the color distribution of the image (similar to a color histogram) and the local spatial structure of the color. The additional information about color structure makes the descriptor sensitive to certain image features to which the color histogram is blind. Figure 13.7 illustrates this with a pair of images, each of which consists of two *iso-color planes*[3], one grey and one black. The grey iso-color plane on the left is highly structured, whereas the one on the right is less so. The *structure* of an iso-color plane is the degree to which its pixels are clumped together relative to the scale of an associated structuring element.

Each image contains exactly 50 pixels in its grey plane and 250 pixels in its black plane. Hence, they are indistinguishable, based solely on the information in their two-bin color histograms. But their two-bin CS Descriptors are very different and thus the images can be easily distinguished in an indexing or retrieval application based on the CSD.

The CSD is identical in form to a color histogram but is semantically different. Specifically, the CSD is a 1D array of eight bit–quantized values,

$$\text{CSD} = \overline{h}_s(m), \quad m \in \{1, \ldots, M\}$$

where M is chosen from the set {256, 128, 64, 32} and where s is the scale of the associated square structuring element. In the example of Figure 13.7, $s = 3^2$. The M bins (array elements) of $\overline{h}_s$ are associated in a bijective[4] manner to the M cells of the nonuniformly quantized HMMD color space (see Section 13.2).

13.6.1 CSD Interoperability

Descriptor interoperability was discussed generally in Section 13.1. There is, however, an aspect of interoperability that is peculiar to the CSD. In retrieval applications it may be

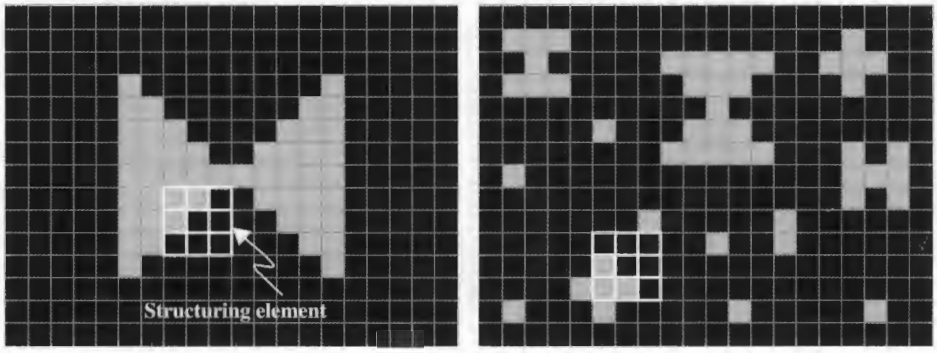


Figure 13.7 Two iso-color planes with differing amounts of structure

[3] An image quantized to N colors is composed of N iso-color planes. The n-th plane is the set of all pixels having the n-th quantized color, $n \in \{1, ..., N\}$.
[4] A map, $f: A \rightarrow B$, is said to be *bijective* if f maps set A onto set B in a one-to-one manner.

the case that a query descriptor presented (e.g. via the Web) to a remote search engine has a length that differs from the descriptors in the database. In order to compute the similarity between query and database descriptors, the lengths must be equalized.

Now in the case of color histograms, a length N histogram can be obtained either

- By extracting it directly from an image quantized to N colors or
- By extracting from the same image, quantized to $M > N$ colors, a length M histogram and then unifying (summing) appropriate subsets of its bin values to form the N-bin histogram.

Either method results in the *same* histogram as long as a *scalability condition* is met by the quantized color space as discussed in [13].

The CSD does not enjoy this property because the color quantization of an image affects its color structure. The reader is directed to [13] for a discussion of the somewhat subtle reason for this. The salient point is that a CS Descriptor obtained from image I by one scheme will, in general, lead to different retrieval results than a CSD from I by the other scheme. That is, the two extraction or resizing methods are not interoperable.

Consequently, and in contrast to most other MPEG-7 visual descriptors, extraction and resizing of the CSD is a *normative* process within the standard, by which we mean that the major steps are specified by the standard. Deviation from these steps risks breaking the interoperability of the descriptor.

13.6.2 Extraction

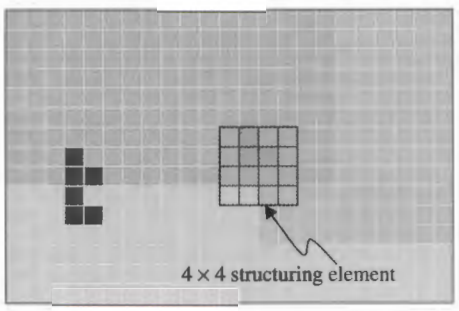
The CSD is best understood in terms of the *Color Structure Histogram*, h_s, upon which $\overline{h}_s$ is based. Extraction of a CSD is a three-step process:

1. A 256-bin CS Histogram is extracted (i.e. accumulated) from an image represented in the 256 cell–quantized HMMD color space. If the image is in another color space, then it must be converted to HMMD and requantized prior to extraction.
2. If $N < 256$ is desired, then bins are unified to obtain a N-bin CS Histogram.
3. The values (amplitudes) of each of the N bins are nonlinearly quantized in accordance with the statistics of color occurrence in typical consumer imagery.

We now discuss these steps in more detail, a full description of which are given in [5] and [14].

Accumulation of CS Histogram

In the context of the CSD, the length and color space of the CS Histogram are fixed. Outside this context, however, the CS Histogram can, in general, be of any length and can be accumulated from an image represented in any quantized color space. The procedure is depicted in Figure 13.8 where a simple five-color 'image' is shown together with a 4×4 structuring element. Also shown in tabular form on the right is an eight-bin CS Histogram,



Color	Bin value
c_1	$h(1)+1$
c_2	$h(2)$
c_3	$h(3)$
c_4	$h(4)+1$
c_5	$h(5)$
c_6	$h(6)+1$
c_7	$h(7)+1$
c_8	$h(8)$

Figure 13.8 (see also Plate 3) Accumulation of CS Histogram

$h_s(m)$, whose bins are associated with eight quantized colors, c_m, $m \in \{1, \ldots, 8\}$, in which the image is represented.

In nominal operation, the structuring element scans the image such that

- The element visits every position in the pixel grid and
- The element always lies entirely within the image.

At each position, the CS Histogram is updated on the basis of the colors present within the element. The operation is illustrated in Figure 13.8 where, in its current position, four colors are present within the structuring element. Therefore, each of the four corresponding bins of the CS Histogram is incremented by one. Observe that in any given position, the increase in $h(m)$ is determined by whether color c_m is present or absent within the element rather than by how much of c_m is enclosed. Hence, the final value of $h(m)$ is determined (up to normalization) by the number of positions at which the structuring element contains c_m.

It is interesting to note that the CS Histogram may be viewed as a *generalized* color histogram since it reduces precisely to an ordinary color histogram when a 1×1 structuring element is used.

Although a 4×4 element is shown in Figure 13.8, the MPEG-7 Standard defines the scale to be 8×8. This was determined by experiment to be the optimal scale. In conjunction with this, the Standard calls for images that deviate from a nominal size to be uniformly subsampled, both horizontally and vertically, in order to reduce the computational load. The subsampling factor is given by $K = 2^p$ where

$$p = \max\{0, \lfloor \log_2 \sqrt{W \cdot H} - 7.5 \rfloor\}$$

where W and H are the picture width and height, respectively, and where $\lfloor \cdot \rfloor$ is the *floor* operator. The reader is directed to References 5 and 14 for an equivalent formulation in which the accumulation requires no explicit subsampling of the image.

The CS Histogram (and hence the CSD) can be extracted from arbitrarily shaped, possibly disconnected, regions of an image. This is done in practice by means of a binary mask that defines the regions. Movement of the structuring element is as above (i.e. over

the entire extent of the image), but the histogram is accumulated only with pixels that lie in the transparent portions of the mask.

Bin unification

When a CSD of length $N \in \{128, 64, 32\}$ is required, the 256-bin CS Histogram is reduced in length by bin unification. This process adds the values in each of N disjoint subsets of bins from the full-length histogram to form the N bins of the shorter histogram.

We now describe the procedure for a general size reduction from M to $N < M$ bins. For the case at hand, one merely lets $M = 256$ and $N \in \{128, 64, 32\}$. Let $P = \{p_1, \ldots, p_M\}$ and $Q = \{q_1, \ldots, q_N\}$ be two scalable quantizations of a color space, S, where the p_m and q_n are the individual cells of the two quantizations and where $M > N$. Quantization *scalability* is equivalent to the conditions:

$$\bigcup_{m=1}^{M} p_m = S = \bigcup_{n=1}^{N} q_m$$

$$\text{for each } n, \qquad q_n = \bigcup_{m \in J_n} p_m, \quad \text{where } J_n = n_1, \ldots, n_{k_n}$$

$$J_i \cap J_j = \emptyset \quad \text{for } i \neq j$$

The first condition insures that both P and Q cover the space, S. The second condition implicitly defines the index subsets, J_n. The third condition is a consequence of the fact that quantization cells are, themselves, disjoint. Hence, it is redundant, following from the second condition. We include it for clarity.

In light of the bijection between bins and color space cells, the bin unification is defined by

$$h_s^N(n) = \sum_{m \in J_n} h_s^M(m), \quad n \in \{1, \ldots, N\} \tag{1}$$

where the superscripts denote the respective histogram lengths. The index subsets, J_n, for reducing a 256 CS Histogram length to a shorter length can be derived from the four scalable quantizations of the HMMD color space defined in Section 13.2.2.

Bin value quantization

The final step in extracting an N-bin CSD is to normalize to the range [0, 1) the bin values (amplitudes) of the N-bin CS Histogram from the preceding step, and then to nonlinearly quantize the normalized values to eight bits according to the quantization table in [5]. The nonlinear quantization was derived using several heuristics and experiments conducted with the CCD and it dramatically increases the retrieval accuracy of the CSD. The chief effect of the nonlinearity is to give the small values greater weight in the similarity measure than they would otherwise have.

13.6.3 CSD Resizing

The extraction procedure of Section 13.6.2 insures that lengths, say N and $M > N$, of two different length CSDs can always be equalized. The resizing procedure adjusts the longer descriptor to match that of the shorter. First, the bin values must be de-quantized so that *linear* values participate in the bin unification. Next, the M bins are unified just as discussed for the general case in Section 13.6.2. Finally, bin values of the new N-bin histogram are nonlinearly requantized to obtain the desired N-bin CSD. It can be shown that this resizing process gives the same result as having extracted an N-bin CSD in the first place.

13.6.4 Retrieval Results

As with other histogram descriptors, the CSD uses the l_1-norm for matching in its Similarity Measure. The CCD was modified by the addition of a few more query images to further differentiate the retrieval performance between the CSD and SCDs. Table 13.6(a) shows CSD retrieval accuracy for the four lengths defined by the standard. The longest descriptors yield the best results.

To motivate the choice of the nonuniformly quantized HMMD color space, Table 13.6(b) lists the retrieval results in the case in which the CS histograms were extracted in the HSV color space followed by nonlinear bin value quantization. The (uniform) color space quantization of the HSV space for each descriptor length is shown in the second column of Table 13.6(b). A comparison of the results in the two tables clearly shows the performance gained by using the nonuniformly quantized HMMD color space.

13.7 COLOR LAYOUT DESCRIPTOR

The color layout descriptor (CLD) is a very compact and resolution-invariant representation of color for high-speed image retrieval. It is designed to efficiently represent spatial distribution of colors. This feature can be used for a wide variety of similarity-based retrieval, content filtering and visualization. It is especially useful for spatial structure–based retrieval applications, for example, sketch-based retrieval and video segment identification. The sketch-based retrieval is considered to be a very important functionality since it can offer very user-friendly interfaces, especially when the search is fast enough.

Table 13.6 CSD retrieval results using a) HMMD color space and b) HSV color space

a)

Descriptor size	ANMRR
256 bins	0.06799
128 bins	0.07613
64 bins	0.09374
32 bins	0.14438

b)

Descriptor size	H × S × V quant.	ANMRR
256 bins	16 × 4 × 4	0.08707
128 bins	8 × 4 × 4	0.09204
64 bins	8 × 2 × 4	0.10700
32 bins	8 × 2 × 2	0.14832

The functionalities of this descriptor are image-to-image matching, video clip–to–video clip matching and sketch-to-image or sketch–to–video clip matching. These functionalities can also be achieved using a combination of the Grid Layout data type of MPEG-7 and the dominant color descriptor. However, this combination would require a relatively large number of bits, and matching will be more complex and expensive. The CLD provides more precise and faster retrieval using more compact description.

13.7.1 Extraction

This descriptor is obtained by applying the DCT transformation on a 2-D array of local representative colors in Y or Cb or Cr color space. Figure 13.9 illustrates the extraction process of the descriptor from an image. It consists of four stages – image partitioning, representative color detection, DCT transformation and nonlinear quantization of the zigzag-scanned coefficients. In the first stage, an input picture is divided into 64 blocks to guarantee the resolution or scale invariance. In the next stage, a single representative color is selected from each block. Any method to select the representative color can be applied, but it is recommended to use the average of the pixel colors in a block as the corresponding representative color since it is most simple and the description accuracy is sufficient in general. The selection results in a tiny image icon of size 8×8. In the third stage, each of the three color components is transformed by 8×8 DCT, so three sets of 64 DCT coefficients are obtained. They are zigzag-scanned and the first few coefficients are nonlinearly quantized (using 64 and 32 levels for DC and AC coefficients, respectively). The standard allows scalable representation of the feature by controlling the number of enclosed coefficients. It is recommended to use a total of 12 coefficients, 6 for luminance and 3 for each chrominance, for most of the images. However, another option to use a total 18 coefficients (6 for both luminance and chrominance) can also be considered to apply this descriptor for high-quality still pictures. The total bit length of the recommended descriptor (12 coefficients) is just 64 bits including one signaling bit, which specifies the extension of the number of coefficients. It should be noted that this descriptor is one of the more compact descriptors in the MPEG-7 visual tools and is quite suitable for applications having limitations on storage and/or bandwidth.

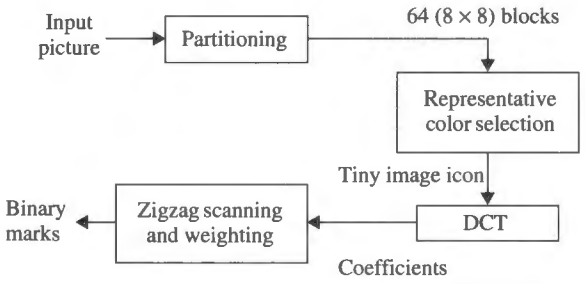


Figure 13.9 The extraction process of the CLD

Representation

The number of DCT coefficients used in the CLD is variable and is represented by the CoefficientPattern field. The CoefficientPattern field can take three possible values. The first value indicates the use of six DCT coefficients for luminance and three each for chrominance, the second values indicates the use of six coefficients for both luminance and chrominance. For the third value of the CoefficientPattern, the number of DCT coefficients are represented by the following NumberofYCoeff and NumberofCCoeff fields. The possible number of coefficients is one of 3, 6, 10, 15, 21, 28 and 64. The actual values of the coefficients are represented by the arrays Ycoeff, CbCoeff and CrCoeff. The lengths of each of these are either five or six bits depending on the coefficient.

Field	Number of bits	Meaning
CoefficientPattern	1–2	Specifies the number DCT coefficients
NumberofYCoeff	3	Number of DCT coefficients for the luminance
NumberofCCoeff	3	Number of DCT coefficients for the chrominance
YCoeff[]	5–6	The DCT coefficients values for the luminance
CbCoeff[]	5–6	The DCT coefficients values for the chrominance
CrCoeff[]	5–6	The DCT coefficients values for the chrominance

13.7.2 Matching

This descriptor is applicable both to an image as a whole and any parts of an image with arbitrary shapes. On applying to an arbitrary-shaped region, the representative color selection should be performed using only valid pixel values, and a padding process is required before the DCT transform. Representative colors of grid blocks containing no valid pixels are substituted with the average color of all valid pixels in the image.

For matching two CLDs, $\{DY, DCr, DCb\}$ and $\{DY', DCr', DCb'\}$, the following distance measure can be used:

$$D = \sqrt{\sum_i w_{yi} (DY_i - DY_i')^2} + \sqrt{\sum_i w_{bi} (DCb_i - DCb_i')^2} + \sqrt{\sum_i w_{ri} (DCr_i - DCr_i')^2}$$

Here, the subscript i represents the zigzag-scanning order of the coefficients. The perceptual characteristic of human vision system could be included for similarity calculation since the feature description is in frequency domain. The distances should be weighted appropriately, with larger weights given to the lower frequency components, to match the characteristic. Since the complexity of the similarity matching process shown above is

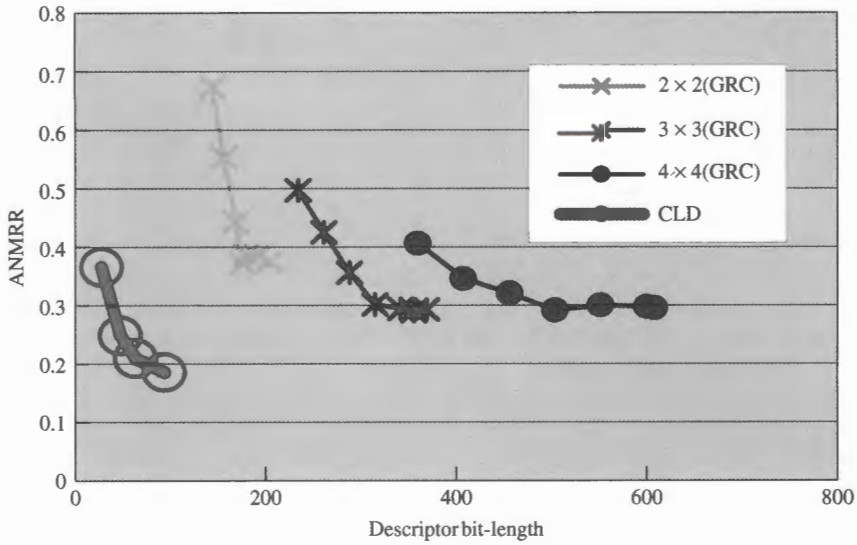


Figure 13.10 The retrieval efficiency of color layout compared with grid-based representative color

low (about 110 clocks with Intel SSE instruction set), super high-speed image matching can be achieved.

13.7.3 Experimental Results

Figure 13.10 shows the retrieval efficiency of this descriptor evaluated using the CCD and Queries. These results demonstrate that the CLD is quite effective in image retrieval in spite of its compact size. The retrieval efficiency is compared with a traditional approach (GRC) wherein the image is partitioned and representative colors for each partition is used to represent the layout feature. The results in Figure 13.10 indicate that CLD achieves a much superior performance than GRC.

Video clip retrieval is one of the more promising applications of the CLD [15]. It requires repetitive use of the matching computation, so high speed matching is necessary to obtain retrievals in a reasonable time. A temporal series of CLDs can be used to implement this functionality. The similarity between video clips is obtained by averaging the distances between corresponding frames of the video clips to be matched.

13.8 SUMMARY

Within the Visual part of MPEG-7, a set of color descriptors has been defined that are able to capture the important aspects of color feature, allowing color similarity computations. These descriptors are compact, hence allowing efficient description of color properties.

Extensive effort, based on fruitful cooperation of a large group of people, has been spent to achieve optimized solutions. It can be expected that the MPEG-7 color descriptors will be extremely useful in those applications that are based on color similarity judgment.

REFERENCES

[1] D. Zier and J.-R. Ohm, "Common datasets and queries in MPEG-7 color core experiments," TR#13-01, October 1999.

[2] H. J. Kim and J. E. Lee, "CE Result of CT1: interoperability between color histogram descriptors using different color spaces and quantization methods," TR#13-03, March 2000.

[3] J.-R. Ohm and B. Makai, "Results of CE CT1 on interoperability of different color histograms," TR#13-04, March 2000.

[4] L. Cieplinski, "Results of Core Experiment CT4 on Dominant Color Extension", TR#13-06, March 2000.

[5] ISO/IEC 15938-3:2001, "Multimedia Content Description Interface – Part 3: Visual," Version 1.

[6] Y. Deng *et al.*, "An efficient color representation for image retrieval," *IEEE Transactions on Image Processing*, **10**(1), 140–147 (2001).

[7] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Norwell, Mass., 1993.

[8] C. Kenney, Y. Deng, B. S. Manjunath and G. Hewer, "Peer group image enhancement," *IEEE Transactions on Image Processing*, **10**(2), 326–334 (2001).

[9] A. Mufit Ferman *et al.*, "Core Experiment on Group-of-Frames/Pictures Histogram Descriptors (CT7)", TR#1305, October 1999.

[10] A. Mufit Ferman *et al.*, "Group-of-Frame/picture color histogram descriptors for multimedia applications", *Proceedings of the IEEE International Conference On Image Processing (ICIP '2000)*, Vol. 1, Vancouver, Canada, 2000, pp. 65–68.

[11] S. Krishnamachari and M. Abdel-Mottaleb, "Hierarchical clustering for fast image retrieval", *Proceedings of Storage and Retrieval for Image and Video Databases VIII*, SPIE Proceedings, 1999, pp. 427–435.

[12] S. Krishnamachari and M. Abdel-Mottaleb, "Image Browsing using Hierarchical Clustering," *Proceedings of the Fourth IEEE Symposium on Computers and Communications (ISCC '1999)*, Red Sea, Egypt, 1999, pp. 301–307.

[13] D. S. Messing, P. van Beek and J. Errico, "Using color and local spatial information to describe images," TR#13-07, January 2001.

[14] ISO/IEC 15938-8:2002, "Multimedia Content Description Interface – Part 8: Extraction and Use of MPEG-7 Descriptions," Version 1.

[15] E. Kasutani and A. Yamada, "The MPEG-7 color layout descriptor: a compact image feature description for high-speed image/video segment retrieval," *Proceedings of International Conference on Image Processing 2001*, October, Thessaloniki, Greece, 2001.

[16] P. Ndjiki-Nya *et al.*, "Subjective Evaluation of the MPEG-7 Retrieval Accuracy Measure (ANMRR)," TR#13-02, May 2000.



This document is available on the DVD

14

Texture Descriptors

Yanglim Choi[1], Chee Sun Won[2], Yong Man Ro[3] and B. S. Manjunath[4]

[1]Samsung Electronics Co, Ltd, Kyungki Do, South Korea, [2]Dongguk University, Seoul, South Korea, [3]Multimedia Group, Information and Communication University, Yusung-gu, Taejon, South Korea, [4]University of California, Santa Barbara, California

14.1 INTRODUCTION

Image texture has emerged as an important visual primitive for searching and browsing through large collections of similar looking patterns. Pictures of water, grass, a bed of flowers or a pattern on a fabric, are examples of image texture. Many natural and artificial objects can be distinguished by their texture. Texture is a region property, as is evidenced by these examples. While it is easy to visualize what one means by texture, there is no universally accepted formal definition of texture. One can think of a texture as consisting of some basic primitives (Julesz coined the term 'textons' to refer to these basic primitives), whose spatial distribution in the image creates the appearance of a texture.

The texture descriptors in MPEG-7 facilitate browsing and similarity retrieval in image and video databases. An image can be considered as a mosaic of homogeneous textures so that these texture features associated with the regions can be used to index the image data. For instance, a user, who is browsing an aerial image database, may want to identify all parking lots from the image collection. A parking lot with cars parked at regular intervals is an excellent example of a textured pattern when viewed from a distance. Similarly, agricultural areas and vegetation patches are other examples of textures commonly found in aerial and satellite imagery. Examples of queries that could be supported in this context could include 'Retrieve all Landsat images of Santa Barbara that have less than 20% cloud cover' or 'Find a vegetation patch that looks like this region'.

To support such image retrieval, an effective representation of texture is needed. There are currently three texture descriptors in MPEG-7: a homogeneous texture descriptor, a texture browsing descriptor and an edge histogram descriptor.

The homogeneous texture descriptor provides a quantitative representation using 62 numbers, consisting of the mean energy and the energy deviation from a set of frequency channels. It is useful for similarity retrieval and as its name indicates, it is quite effective in characterizing homogeneous texture regions.

In addition to the homogeneous texture descriptor a descriptor is defined for coarse level texture browsing. This descriptor, referred to as the texture browsing descriptor, is very compact and requires a maximum of 12 bits. It provides a perceptual characterization of texture, similar to a human characterization, in terms of regularity, coarseness and directionality of the texture pattern. Since the browsing descriptor relates closely to human characterization, it can also be manually instantiated.

The edge histogram descriptor captures spatial distribution of edges in an image. It is thus useful in matching regions with partially varying, that is, nonuniform texture. Its effectiveness is demonstrated on image data that are not necessarily homogeneously textured, for example, natural images, sketch images and clip art images.

The following sections discuss the three texture descriptors in detail.

14.2 HOMOGENEOUS TEXTURE DESCRIPTOR

The homogeneous texture descriptor (HTD) characterizes the region texture using the mean energy and the energy deviation from a set of frequency channels. The 2-D frequency plane is partitioned into 30 channels as shown in Figure 14.1. The mean energy and its deviation are computed in each of these 30 frequency channels (in the frequency domain). Note that the frequency plane partitioning is uniform along the angular direction (equal

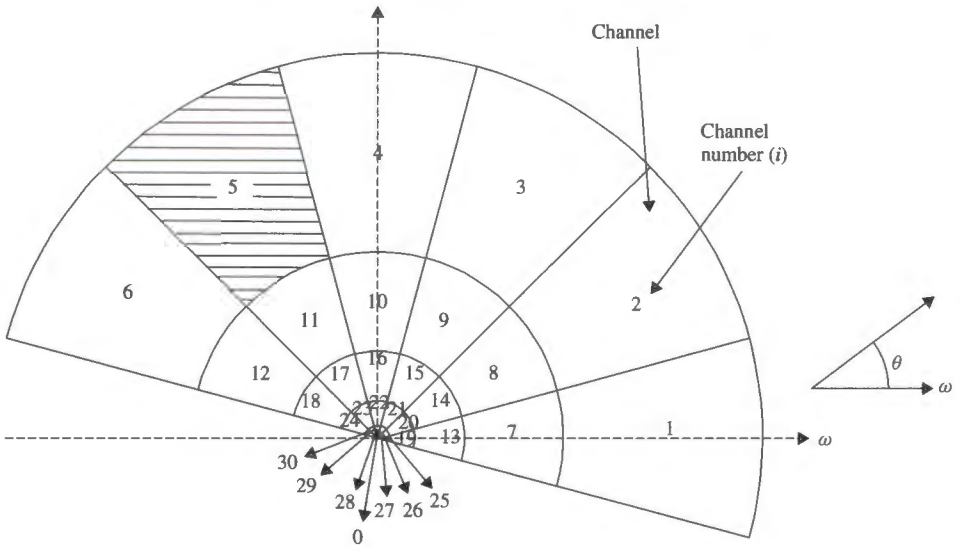


Figure 14.1 Channels used in computing the HTD

See demonstrations of using homogeneous texture descriptor for image search on the DVD.

step size of 30 degrees) but not uniform along the radial direction. More specifically, the division along the radial direction is on an octave scale. The motivation for such a partitioning comes from some of the earlier work. See, for example, [1–5]. There is also some evidence that the early visual processing in the human visual cortex can be modeled well using the frequency layout similar to Figure 14.1. Similar frequency layouts have also been used in image texture analysis [6].

The syntax of the HTD is as follows:

$$\text{HTD} = [f_{\text{DC}}, f_{\text{SD}}, e_1, e_2, \ldots, e_{30}, d_1, d_2, \ldots, d_{30}] \tag{1}$$

Where f_{DC} and f_{SD} are the mean and standard deviation of the image, respectively, and e_i and d_i are the nonlinearly scaled and quantized mean energy and energy deviation of the corresponding ith channel in Figure 14.1, respectively.

14.2.1 Extraction

The first two components of the HTD, the mean and the standard deviation of the input image, are straightforward to compute. The remaining components are computed in the frequency domain directly.

The individual channels in Figure 14.1 are modeled using Gabor functions [7]. Consider a channel indexed by (s, r) where s is the radial index and r is the angular index. Then the (s, r)-th channel is modeled in the frequency domain as:

$$G_{s,r}(\omega, \theta) = \exp\left[\frac{-(\omega - \omega_s)^2}{2\sigma_s^2}\right] \cdot \exp\left[\frac{-(\theta - \theta_r)^2}{2\tau_r^2}\right] \tag{2}$$

where σ_s and τ_r are the standard deviation of the Gaussian in the radial direction and the angular direction, respectively. In the normalized frequency space ($0 \leq \omega \leq 1$), the center frequencies of the feature channels are spaced equally by 30 degrees in angular direction such that $\theta_r = 30^\circ \times r$, where r is the angular index with $r \in \{0, 1, 2, 3, 4, 5\}$. The angular width of all feature channels is 30 degrees. In the radial direction, the center frequencies of the feature channels are spaced by an octave scale such that $\omega_s = \omega_0 \cdot 2^{-s}$, $s \in \{0, 1, 2, 3, 4\}$, where s is the radial index and ω_0 is the highest center frequency specified by $3/4$. The octave bandwidth of the feature channels in the radial direction is written as $B_s = B_0 \cdot 2^{-s}$, $s \in \{0, 1, 2, 3, 4\}$, where B_0 is the largest bandwidth specified by $1/2$.

Figure 14.2 shows the half-power contours of neighboring channels touching each other. Using this as the design criterion leads to the following parameter values:

$$\tau_r = 15^\circ / \sqrt{2 \ln 2} \; (r \in \{0, 1, 2, 3, 4, 5\}) \tag{3}$$

and

$$\sigma_s = B_s / 2\sqrt{2 \ln 2} \tag{4}$$

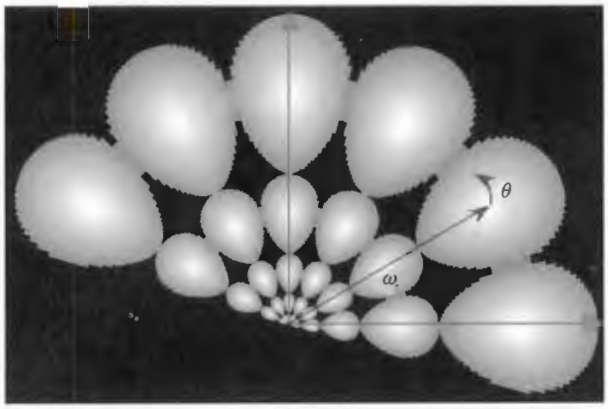


Figure 14.2 Texture feature channels modeled using the Gabor functions in the polar frequency domain

where $B_s = 2^{-(s+1)}$ ($s \in \{0, 1, 2, 3, 4\}$) is the octave bandwidth. We refer to Reference 7 for more details on the design of these filters.

Figure 14.2 shows the extent of the Gabor functions in the 2-D polar frequency plane with six partitions in the angular direction and five in the radial direction.

On the basis of the frequency layout and the Gabor functions, the energy e_i of the ith feature channel is defined as the log-scaled sum of the square of the Gabor-filtered Fourier transform coefficients of an image:

$$e_i = \log_{10}[1 + p_i] \tag{5}$$

where

$$p_i = \sum_{\omega=0^+}^{1} \sum_{\theta=(0°)^+}^{360°} [G_{s,r}\ (\omega, \theta)|\omega|P(\omega, \theta)]^2 \tag{6}$$

and $P(\omega, \theta)$ is the Fourier transform of an image represented in the polar frequency domain, that is, $P(\omega, \theta) = F(\omega\cos\theta, \omega\sin\theta)$, where $F(u, v)$ is the Fourier transform in the Cartesian coordinate system. The integration is over the entire frequency layout (the radial direction is normalized to (0,1) except the DC part. Note that $i = 6 \times s + r + 1$. The energy deviation d_i of the ith feature channel is defined as the log-scaled standard deviation of the square of the Gabor-filtered Fourier transform coefficients of an image:

$$d_i = \log_{10}[1 + q_i] \tag{7}$$

where

$$q_i = \sqrt{\sum_{\omega=0^+}^{1} \sum_{\theta=(0°)^+}^{360°} \{[G_{s,r}\ (\omega, \theta)\ |\omega|P(\omega, \theta)]^2 - p_i\}^2} \tag{8}$$

Finally, the HTD consists of the mean and standard deviation of the image intensity, the energies e_i, and (optionally) the energy deviations d_i of the feature channels.

The above operations can be efficiently performed using the Radon transform [8]. The Radon transform is defined as the integral along the line that has an angle θ counter-clockwise from the y-axis and at a distance R from the origin. It can be written as

$$p_\theta(R) = \int_{L(R,\theta)} f(x, y)\mathrm{d}l = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)\delta(x\cos\theta + y\sin\theta - R)\mathrm{d}x\mathrm{d}y \quad (9)$$

where $f(x, y)$ is an image function, R is the projection axis and $\delta(\cdot)$ is the delta function. The *projection slice theorem* [8] guarantees that the 1D Fourier transform of a projection of image at angle θ equals the slice at angle θ through the 2-D Fourier transform of that image. The $P(\omega, \theta)$ and hence, the filtered images can thus be efficiently computed using the Radon transform, and details can be found in [9].

The components of the descriptor are nonlinearly quantized to eight bits per component. The quantization details can be found in the ISO text [10].

14.2.2 Experiments and Applications

The similarity distance between two texture images is measured by summing the weighted absolute difference between two sets of feature vectors, one for a query image (TD_{query}) and the other in the database (TD_{Database}).

Similarity is measured by calculating the distance between two feature vectors, and is given by

$$\begin{aligned} d(TD_{\text{query}}, TD_{\text{Database}}) &= \text{distance}(TD_{\text{query}}, TD_{\text{Database}}) \\ &= \sum_k \left| \frac{TD_{\text{query}}(k) - TD_{\text{Database}}(k)}{\alpha(k)} \right| \end{aligned} \quad (10)$$

The recommended normalization value $\alpha(k)$ is the standard deviation of $TD_{\text{Database}}(k)$ for a given database but the user can specify his own choice of $\alpha(k)$, for example, $\alpha(k) = 1$.

For intensity invariance matching, the first component of the feature vector (i.e. the average intensity of the image) is not used in computing the dissimilarity.

For rotation invariance, recall that the frequency space division for the texture feature is made in the polar frequency domain. The texture descriptor of a rotated image is an angular-shifted version of the reference image. We first measure the distance between texture vectors in the database (TD_{Database}) and a query texture vector (TD_{query}) by shifting the query texture vector in the angular direction such as,

$$d(TD_{\text{query}}, TD_{\text{Database}}, m\phi) = \text{distance}(TD_{\text{query}}|_{m\phi}, TD_{\text{Database}}) \quad (11)$$

where $\phi = 30$ *degrees*. Then, for rotation-invariant matching the distance is calculated as [11],

$$d(TD_{\text{query}}, TD_{\text{Database}}) = \arg\min_m \{d(TD_{\text{query}}, TD_{\text{Database}}, m\phi) | m = 0, \ldots, 5\} \quad (12)$$

Similarly, one can perform scale invariant matching though the effectiveness there is quite limited because of the coarse sampling in scale-space.

The performance of this texture descriptor is evaluated on a large texture image data set consisting of images from Brodatz album in [12], aerial images in [13] and stock photo images and textures from Corel. The total number of images used in the core experiments exceeds 10 000. On the Brodatz data set, with experimental conditions as described in Reference 9, the average retrieval accuracy is about 77%.

In the Alexandria Digital Library (ADL) project at the University of California, Santa Barbara, this texture descriptor is being used to annotate a large collection of aerial imagery [13]. Remotely sensed data, such as aerial imagery, is an excellent data set on which to demonstrate the application potential of the descriptor. The texture descriptor is extracted from near 1-m resolution aerial images of the Santa Barbara, California region. Each aerial image is more than 5000×5000 pixels in size. The descriptor is extracted from nonoverlapping 128×128 pixel tiles resulting in more than 1600 descriptor instances per aerial image. The descriptor supports effective nearest-neighbor and range similarity searches. A demonstration is available on-line at the UCSB Vision Research Lab Web site (*http://vision.ece.ucsb.edu*). The data set for this demonstration consists of more than 50 aerial images or approximately 100 000 tiles. Figure 14.3 shows the results of a nearest-neighbor similarity search using the texture descriptor extracted from an agricultural region.

In the ADL the researchers are using the texture descriptor to classify regions extracted from large collections of aerial imagery. This classification permits users to

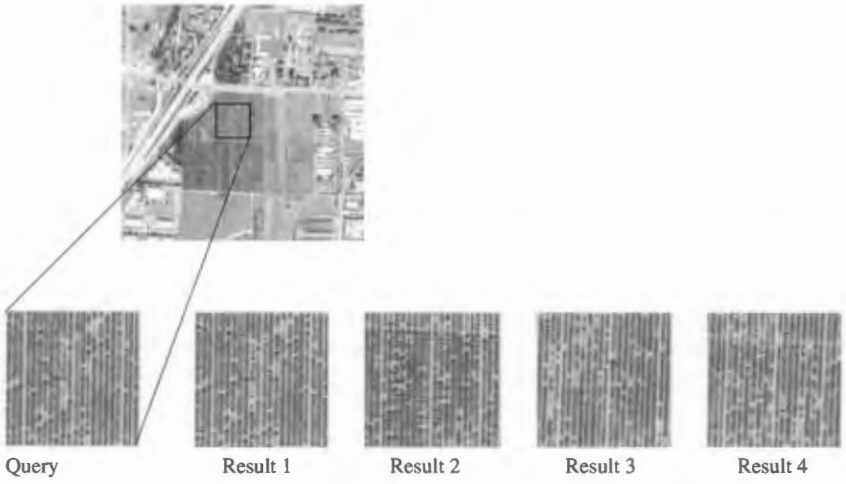


Figure 14.3 Search results using the HTD on an aerial image database

interact with the collections at a more meaningful level. A promising direction for this work combines the discriminating capability of the texture descriptor with the land-type labels from an extensive gazetteer in a supervised learning framework.

14.3 TEXTURE BROWSING DESCRIPTOR

14.3.1 Definition and Semantics

The texture browsing descriptor (TBD) specifies the perceptual characterization of a texture, which is similar to a human characterization, in terms of regularity, coarseness and directionality. This representation is useful for browsing applications and coarse classification of textures. Its format and semantics are as follows:

$$\text{TBD} = [v_1\ v_2\ v_3\ v_4\ v_5] \tag{13}$$

$v_1 \in \{1, 2, 3, 4\}$: Represents how regular or structured the texture is. The larger the value of v_1, the more regular is the texture. Figure 14.4 shows four textures with different requlrity values.

$v_2, v_3 \in \{1, \ldots, 6\}$: Two quantized directions that best capture the directionality of the texture.

$v_4, v_5 \in \{1, \ldots, 4\}$: Two quantized scales that best capture the coarseness of the texture. A larger value indicates a coarser texture.

14.3.2 Extraction

A schematic of the TBD computations is shown in Figure 14.5. Filtering is performed using the Gabor filter extraction method with a similar frequency layout as in the HTD (Section 14.2, Figure 14.1). From the multiresolution decomposition, a given image is decomposed into a set of filtered images. Each of these images represents the image information at a certain scale and at a certain orientation. Its computation is based on the following observations:

- Structured textures usually consist of dominant periodic patterns.
- A periodic or repetitive pattern, if it exists, can be captured by the filtered images. This behavior is usually captured in more than one filtered output.
- The dominant scale and orientation information can also be captured by analyzing projections of the filtered images.

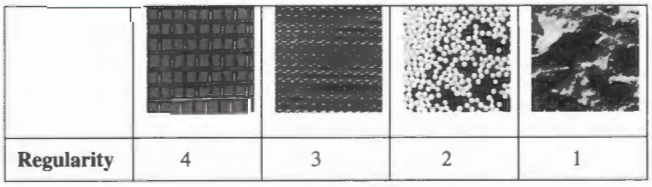


Figure 14.4 Texture Regularity

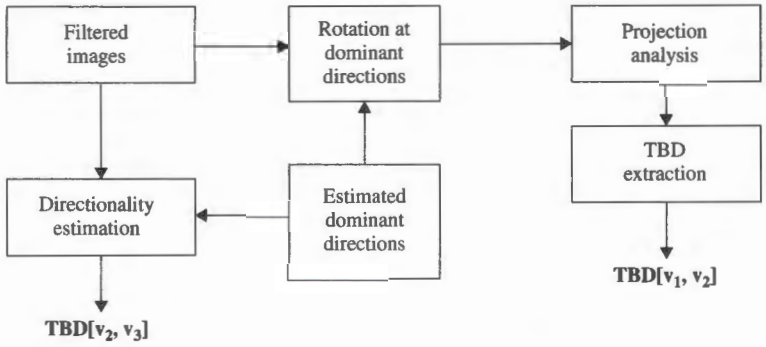


Figure 14.5 Extraction of the TBD

Dominant direction estimation

The directionality can be estimated either in spatial domain [14] or in frequency domain [15, 16]. Because of the aperture effect [16], the spatial domain approaches seem to be more suitable for directionality detection [17]. The approach described here is based on directional histograms computed from the filtered images. The $S \times K (S = 4, K = 6)$ filtered images are used to construct S directional histograms,

$$H(s,k) = \frac{N(s,k)}{\sum_{k=0}^{k=5} N(s,k)}, \quad s = 0, \ldots, 3 \text{ and } k = 0, \ldots, 5 \tag{14}$$

where $N(s,k)$ is the number of pixels in the filtered image at scale s and direction k whose magnitude is larger than a threshold t_s[1]. Note that $\theta_k = \frac{k}{K} \times \pi$ radians. The locations of the peaks in the histogram are indicators of the dominant directions. Now the direction(s), in which there are peaks in the neighboring scales, are chosen as the dominant direction(s). It is based on the assumption that if a direction is dominant at one scale, it should also be dominant at a neighboring scale. Among these dominant directions, the directions corresponding to the two peaks with the highest sharpness are chosen as TBD[v_2] and TBD[v_3]. The sharpness of a peak $H(s,k)$, is defined to be $C(s,k) = 0.5 \cdot [2H(s,k) - H(s,k-1) - H(s,k+1)]$. It is an indication of the faithfulness of the estimation [17].

14.3.3 Regularity and Coarseness Estimation

For each of the $S \times K$ filtered images $W_{mn}(x,y)$, two projections, denoted as $P_H^{(mn)}$ and $P_v^{(mn)}$, are computed using the Radon transform

$$P_H^{(mn)}(l) = \iint W_{mn}(x,y)\delta(x\cos\theta_{DO1} + y\sin\theta_{DO1} - l)\mathrm{d}x\mathrm{d}y \tag{15}$$

$$P_V^{(mn)}(l) = \iint W_{mn}(x,y)\delta(x\cos\theta_{DO2} + y\sin\theta_{DO2} - l)\mathrm{d}x\mathrm{d}y \tag{16}$$

[1] At each scale s, the threshold is set to $t_s = \mu_s + \sigma_s$, where μ_s and σ_s are the mean and the standard deviation over the all K filtered images at that scale s.

where θ_{DO1} and θ_{DO2} are the two dominant directions estimated as in the previous section. Consider the projection corresponding to θ_{DO1}, (a similar analysis will also be performed on the projection corresponding to θ_{DO2}.) and let $P(l)$ be a projection in this set (dropping the subscripts for notational convenience.), the Normalized Autocorrelation Function (NAC) is computed for each of these projections as

$$\mathrm{NAC}(k) = \frac{\sum_{m=k}^{N-1} P(m-k)P(m)}{\sqrt{\sum_{m=k}^{N-1} P^2(m-k) \sum_{m=k}^{N-1} P^2(m)}} \tag{17}$$

The local peaks and valleys of NAC(k) and their positions and magnitudes are recorded. Let M be the number of peaks and N be the number of valleys, let $p\_posi(i)$, $p\_magn(i)(i = 1, 2, \ldots, M)$ be the positions and magnitudes of these peaks and $v\_posi(j)$, $v\_magn(j)(j = 1, 2, \ldots, N)$ be the positions and magnitudes of the valley points, then the contrast is defined as

$$contrast = \frac{1}{M}\sum_{i=1}^{M} p\_magn(i) - \frac{1}{N}\sum_{j=1}^{N} v\_magn(j) \tag{18}$$

Given a peak sequence $p\_posi(i)$, the average of the distances among the successive peaks, *dis*, and the standard deviation of distances, *std* are computed. Let $\gamma = std/dis$. If γ is smaller than a preselected threshold T_p, the corresponding projection is considered to be a candidate. Simple *agglomerate clustering* in the two-dimensional $std - dis$ space for the candidate projections is then used to remove the outliers. From the resulting candidates, we identify the ones with the maximum contrast. Let $m^*(H)$ denote the scale index of $P_H^{(mn)}$ with the maximum contrast and similarly, let $m^*(V)$ denote the scale of $P_v^{(mn)}$ with maximum contrast, then we set

$$\mathrm{TBD}[v_4] = m^*(H) \text{ and } \mathrm{TBD}[v_5] = m^*(V).$$

The regularity estimate is obtained by assigning credits to the candidate projections. Let the candidate set be denoted by Ω. Each of the projections in Ω gets a certain credit depending on whether its neighboring projection (in scale-orientation space) is also in Ω or not. The credits from all the projections in Ω are then added together and quantized to be an integer in the range of [1, 4]. A value of 4 indicates strong regularity and a value of 1 implies an irregular or unstructured texture. For more details see [18].

14.3.4 Experiments and Applications

Figure 14.6 shows an example of dominant orientation estimation. Four quadrants of the texture T069 from the Brodatz album and their corresponding directionalities are shown. As reported in Reference 17, T069 is a texture pattern that is difficult to analyze because

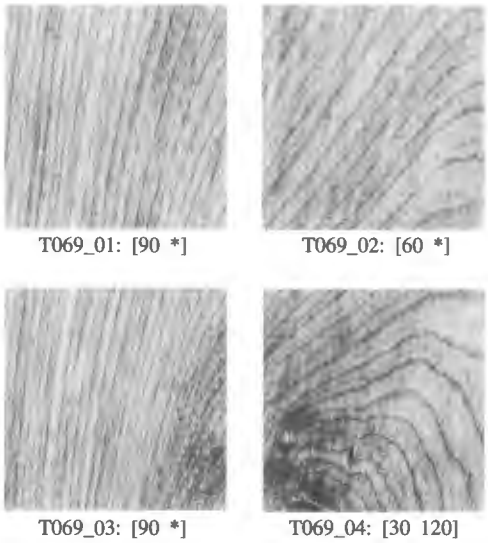


Figure 14.6 Detection of gradual change in texture orientation

of the gradual change in directionality. Note that the estimated orientation for each of the patterns in Figure 14.6 match reasonably well with the perceived directionality of the texture.

Figure 14.7 shows few more texture images and their associated estimated dominant direction(s). In general, it is observed that the proposed method for dominant orientation estimation works quite well in coarsely quantizing the texture orientation. One limitation of this method is that it does not capture well the overall macropattern orientation of the texture if the texture micropatterns have a significantly different directionality than the macropatterns they form (see, e.g. T036 in Figure 14.7). The reason is that the used filters do not capture the large-scale properties of such texture patterns.

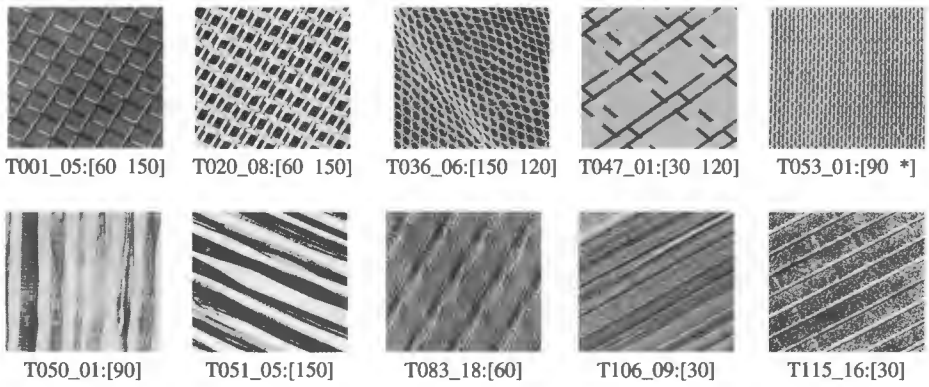


Figure 14.7 Some directional images and their detected dominant direction(s). The images in the first row are the ones with two dominant directions recognized. The images in the second row are identified as having only one dominant direction

Subjective tests indicate that the proposed method for computing the three attributes – directionality, coarseness and regularity – compares quite well with the human observations. For details, see Reference 18.

14.4 EDGE HISTOGRAM DESCRIPTOR

14.4.1 Definition and Semantics

Spatial distribution of edges in an image is another useful texture descriptor for similarity search and retrieval. The edge histogram descriptor (EHD) in MPEG-7 represents local-edge distribution in the image. Specifically, dividing the image space into 4×4 subimages as shown in Figure 14.8, the local-edge distribution for each subimage can be represented by a histogram. To generate the histogram, edges in the subimages are categorized into five types; vertical, horizontal, 45° diagonal, 135° diagonal and nondirectional edges. Since there are 16 subimages, a total of $5 \times 16 = 80$ histogram bins are required. Table 14.1 summarizes the semantics of the 80-bin EHD.

14.4.2 Extraction

For the EHD, it is required to detect nondirectional edges as well as four directional edges. The nondirectional edges include the edges with no particular directionality. The five types of edges can be extracted by a block-based edge extraction scheme. To do that, as shown in Figure 14.8, each subimage is further divided into nonoverlapping square image-blocks. The size of the image-block is dependent on the image resolution. Regardless of the size of the given image, the image space is divided into a predetermined number of image-blocks. Experiments show that the number of image-blocks around 1100 seems to capture good directional edge features.

Each of the image-blocks is then classified into one of the five edge categories mentioned above or as a nonedge block. A simple method to do this classification is to

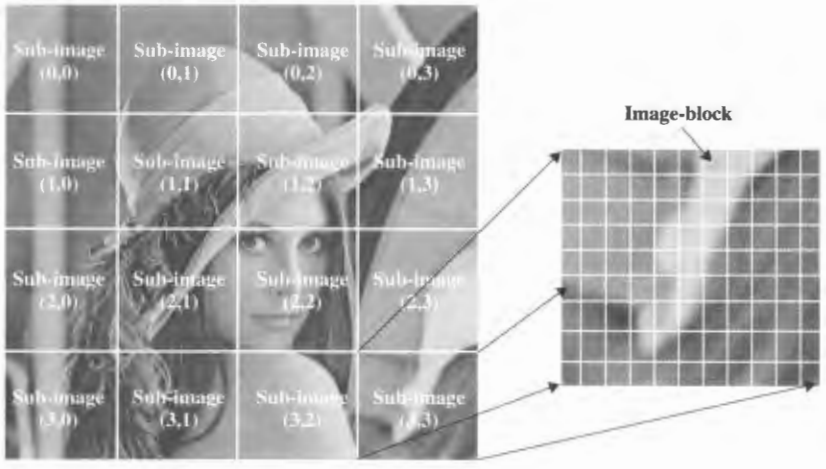


Figure 14.8 Definition of subimage and image-blocks

Table 14.1 Semantics of the histogram bins of the EHD

H_E	Semantics
h(0)	Relative population of vertical edges in subimage at (0,0)
h(1)	Relative population of horizontal edges in subimage at (0,0)
h(2)	Relative population of 45° edges in subimage at (0,0)
h(3)	Relative population of 135° edge in subimage at (0,0)
h(4)	Relative population of nondirectional edges in subimage at (0,0)
⋮	⋮
h(75)	Relative population of vertical edges in subimage at (3,3)
h(76)	Relative population of horizontal edges in subimage at (3,3)
h(77)	Relative population of 45° edges in subimage at (3,3)
h(78)	Relative population of 135° edges in subimage at (3,3)
h(79)	Relative population of nondirectional edges in subimage at (3,3)

treat each image-block as a 2×2 super-pixel image-block and apply appropriate oriented edge detectors to compute the corresponding edge strengths. The edge detector with the maximum edge strength is then identified. If this edge strength is above a given threshold, then the corresponding edge orientation is associated with the image-block. If the maximum of the edge strengths is below the given threshold, then that block is not classified as an edge block. For more detail, we refer the readers to References 19 and 20.

14.4.3 Quantization

Following the classification of each image-block, the five-bin-histogram for each subimage is generated by counting each edge type for all image-blocks in the subimage. Then, each bin value is normalized by the total number of image-blocks in the subimage. Since there are image-blocks without any edge (i.e. nonedge blocks) as well as edge blocks, the sum of five normalized histogram bins for each subimage is less than or equal to 1. This implies that the information regarding nonedge distribution (i.e. monotone regions) in the subimage is also indirectly considered in the EHD.

Now, the normalized bin values are quantized for binary representation. Since most of the values are concentrated within a small range (for instance, from 0 to 0.3), they are nonlinearly quantized to reduce the number of bits. In the MPEG-7 specification, each bin is quantized to 3 bits. Therefore, the total number of bits required for the EHD is 240 (80×3) bits.

14.4.4 Matching

Recall that the 80 bins of the local-edge histogram in Table 14.1 are the standardized normative semantics for the EHD. However, the local-edge histograms alone may not be sufficient for an effective image matching. Instead, some global-edge distributions as well as the local ones are used. Specifically, edge distribution information for the whole image space and some horizontal and vertical semiglobal-edge distributions as well as local ones are required to improve the matching performance. The global-edge histogram

and semiglobal-edge histograms are directly computed from the 80 local histogram bins $h(i)$, $i = 0, \ldots, 79$. For the global-edge histogram, the five types of edge distributions for all subimages are accumulated. Similarly, for the semiglobal-edge histograms, subsets of subimages are grouped as shown in Figure 14.9. In this case, there are 13 different segments (i.e. 13 different subsets of the image-blocks). The corresponding edge histograms for each segment are then generated using the local-edge histograms. Combining the local, the semiglobal and the global histograms together, one can construct a total of 150 bins [80 bins (local) + 5 bins (global) + 65 bins (13 × 5, semiglobal)] for similarity matching.

For similarity matching, the l_1 distance measure D(A,B) can be adopted for two image histograms A and B as in the following equation:

$$D(\mathrm{A}, \mathrm{B}) = \sum_{i=0}^{79} |h_{\mathrm{A}}(i) - h_{\mathrm{B}}(i)| + 5 \times \sum_{i=0}^{4} |h_{\mathrm{A}}^{\mathrm{g}}(i) - {}_{\mathrm{B}}^{\mathrm{g}}\, h(i)| + \sum_{i=0}^{64} |h_{\mathrm{A}}^{\mathrm{S}}(i) - h_{\mathrm{B}}^{\mathrm{S}}(i)| \quad (19)$$

where $h_{\mathrm{A}}(i)$ and $h_{\mathrm{B}}(i)$ represent the normalized histogram bin values of image A and image B, respectively. $h_{\mathrm{A}}^{\mathrm{g}}(i)$ and $h_{\mathrm{B}}^{\mathrm{g}}(i)$ represent the normalized bin values for the global-edge histograms of image A and image B, respectively, which are obtained from the corresponding local histograms $h_{\mathrm{A}}(i)$ and $h_{\mathrm{B}}(i)$. Similarly, $h_{\mathrm{A}}^{\mathrm{S}}(i)$ and $h_{\mathrm{B}}^{\mathrm{S}}(i)$ represent the histogram bin values for the semiglobal-edge histograms of image A and image B, respectively. Since the number of bins of the global histogram is relatively smaller than that of local and semiglobal histograms, a weighting factor 5 is applied in (19).

14.4.5 Experiments and Applications of the EHD

Experimental results (conducted during the MPEG-7 evaluation period) with 11 639 images show that the EHD is quite useful for image retrieval, especially for natural images with nonuniform textures and clip art images. Figures 14.10 and 14.11 are examples of the retrieval results for natural and clip art query images, respectively. The EHD can be also used for image retrieval with a sketch example. Figure 14.12 shows an example of

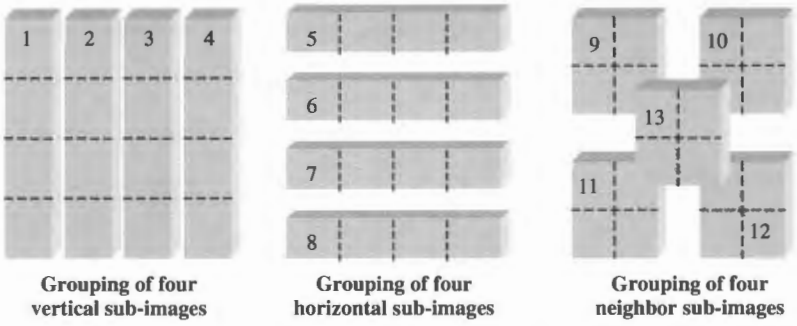


Figure 14.9 Segments of subimages for semiglobal histograms



Figure 14.10 Retrieval result for a natural image query

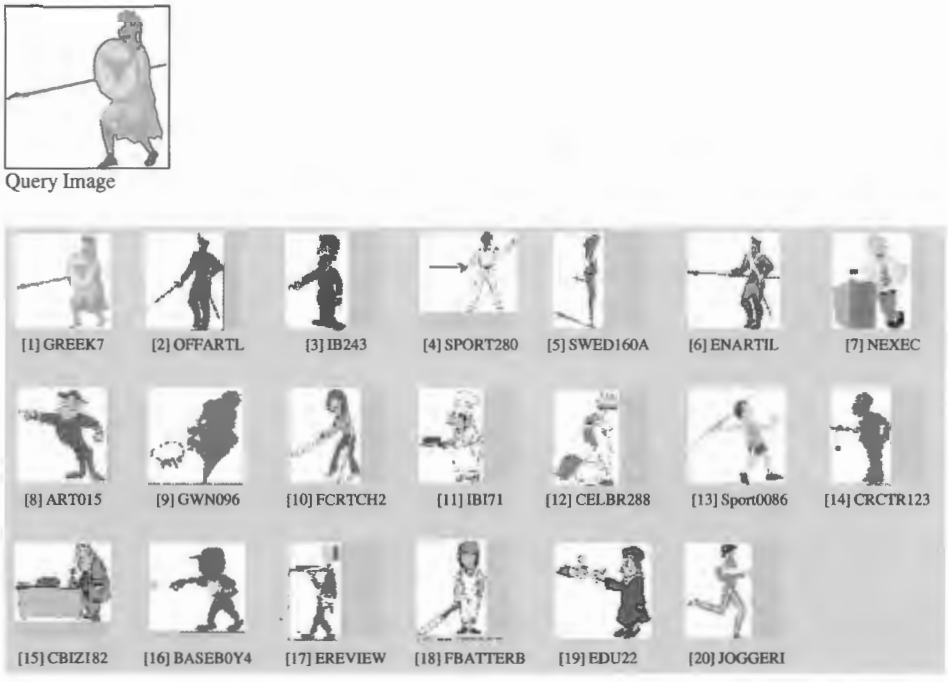


Figure 14.11 Retrieval result for a clip art image

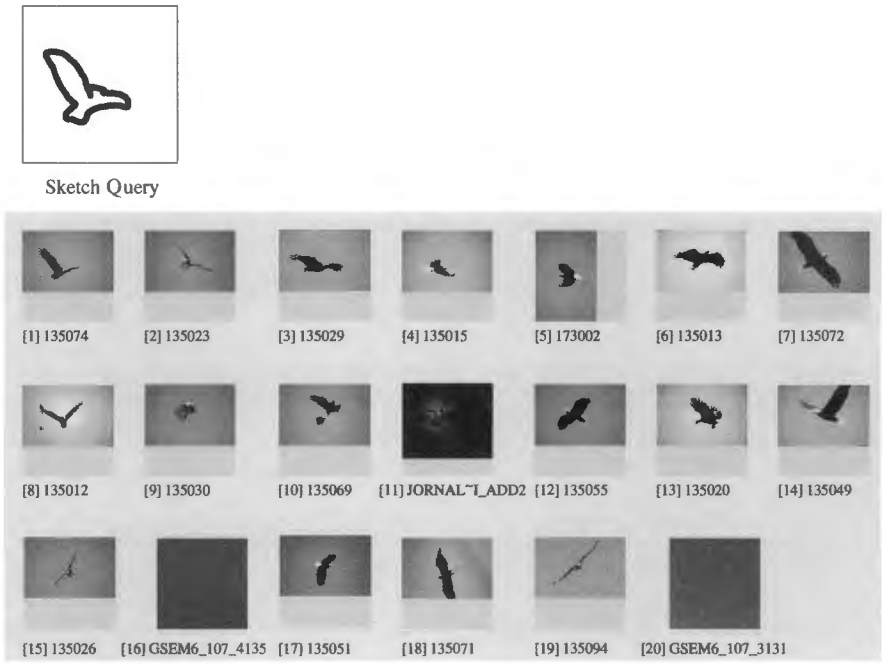


Figure 14.12 Retrieval result for a query by sketch

the query results by the sketch example. For all experiments, the total number of image-blocks to determine the size of the image-block is fixed as 1100. For more examples of the experiments, we refer the readers to References 20 and 21.

The EHD is a compact descriptor with a fixed length. It always takes 240 bits for each image. The EHD is also very flexible, because it consists of local-edge histograms only. Using them, one can generate semiglobal and global-edge histograms with various segments of the image.

14.5 SUMMARY

Texture is a salient feature in images and videos of natural scenes. In the current release of the MPEG-7, there are three descriptors for the texture feature. These descriptors have been extensively tested during the evaluation phase of the standard.

Note that all three descriptors can be used for characterizing arbitrary shaped objects as well as rectangular image frames. In the case of HTD the object is assumed to have a homogeneous texture, so a sample texture of an inscribing rectangle for the object can be taken, with some padding if the sample is too small, to compute the descriptor. For the TBD, the object is also assumed to have a homogeneous texture and the same method as in HTD can be applied. For EHD, a rectangular image obtained from the bounding box of the object with the background padded by a constant grey-level (for instance, zero value) can be used.

Table 14.2 Summary of texture descriptors

	HTD	EHD	TBD
Number of components (number of bits)	62 (496 bits)	80 (240 bits)	5 (12 bits)
Used features	Gabor-filtered responses of frequency domain.	Edges distribution in subimages.	Projections of Gabor-filtered responses in spatial domain
Application domain	Image/video retrieval – Query-by-example	Image/video retrieval – Query-by-example	Image/video browsing – Query-by-example or browsing
Image Category and Examples (Recommended)	Homogeneously textured images Pattern on fabrics Aerial imagery Internet images Trademark images	Nonhomogeneous region Natural images Sketch images Clip art images	Homogeneously textured images Textile/fabric patterns

Table 14.2 summarizes these descriptors and their potential application domains.

As is evident from the table the descriptors provide complementary functionalities, thus, allowing the user to choose the descriptor(s) that best fit the intended application. What has not been actually tested yet is the combination of the three descriptors in a generic texture based image-search engine. It is reasonable to speculate that a carefully designed description scheme would allow the user a greater flexibility to formulate more complex queries that the individual descriptors alone are not capable of handling. A simple weighting scheme using the three descriptors would be a good start. However, these issues need further investigation.

ACKNOWLEDGMENTS

Y. Choi thanks Hyundoo Shin and Yunsoo Kim for their support of this work. CSW thanks Dong Kwon Park for providing experimental results for the EHD. BSM was supported in part by grants from Samsung Electronics Co. and an NSF award #97-04785. He also thanks Peng Wu, Shawn Newsam and Lei Wang for their help in writing this chapter.

REFERENCES

[1] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biological Cybernetics*, **61**, 103–113 (1989).

[2] J. G. Daugman, "High confidence visual recognition of persons by a test of statistical independence," *IEEE Trans.* Pattern Analysis and Machine Intelligence, **15**(11), 1148–1161 (1993).

[3] A. Saadane, H. Senane and D. Barba, "On the design of psychovisual quantizers for a visual subband image coding," *Proceedings of SPIE*, Vol. 2308, 1994, pp. 1446.

[4] A. Saadane, H. Senane and D. Barba, "An entirely psychovisual based subband image coding scheme," *Proceedings of SPIE*, Vol. 2501, pp. 1702, 1995.

[5] Y. M. Ro and K. W. Yoo, "Texture featuring and indexing using matching pursuit in radon space," *Proceedings of IEEE International Conference Image Processing*, Vol. 2, Kobe, Japan, 1999, pp. 580–584.

[6] G. M. Haley and B. S. Manjunath, "Rotation-invariant texture classification using a complete space-frequency model," *IEEE Transaction on Image Processing*, **8**(2), 255–269 (1999).

[7] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**(8), 837–842 (1996).

[8] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, N.J., 1989.

[9] Y. M. Ro, M. C. Kim, H. K. Kang, B. S. Manjunath and J. W. Kim, "MPEG-7 Homogeneous Texture descriptor," *ETRI Journal*, **23**(2), 41–51 (2001).

[10] ISO/IEC 15938-3:2001, "Multimedia content description interface – Part 3: Visual," Version 1.

[11] Y. M. Ro and H. K. Kang, "Hierarchical rotational invariant similarity measurement in MPEG-7 homogeneous texture descriptor," *IEE Electronic Letter*, **36**(15), 1268–1270 (2000).

[12] P. Brodatz, *Textures: A photographic Album for Artists & Designers*, Dover Publications, New York, 1966.

[13] W. Y. Ma and B. S. Manjunath, "A texture thesaurus for browsing large aerial photographs," *Journal of American Society for Information Science*, **49**(7), 633–648 (1998).

[14] H. Tamura, S. Mori and T. Yamawaki, "Texture features corresponding to visual perception," *IEEE Trans., Syst., Man, Cyber*, **SMC8**(6), 460–473 (1978).

[15] F. Liu and R. W. Picard, "Periodicity, Directionality, and Randomness: Wold Features for Image Modeling and Retrieval," MIT Media Lab Technical Report No. 320, March 1995.

[16] J. S. Weszka, C. R. Dyer and A. Rosenfeld, "A comparative study of texture measures for terrain classification", *IEEE Trans., Syst., Man, Cyber.*, **SMC-6**, 269–285 (1976).

[17] A. Branca, M. Tafuri, G. Attolica and A. Distante, "Directionality Detection in Compositional Textures," *Proceedings of International Conference on Pattern Recognition*, Vol. II, 1996, pp. 830–834.

[18] P. Wu, B. S. Manjunath, S. Newsam and H. D. Shin, "A Texture Descriptor for Browsing and Similarity Retrieval," *J. Signal Processing: Image Communication*, **16**, 33–43 (2000).

[19] C. S. Won and D. K. Park, "Image block classification and variable block size segmentation using a model-fitting criterion," *Optical Engineering*, **36**(8), 2204–2209 (1997).

[20] D. K. Park, Y. S. Jeon, C. S. Won and S.-J. Park, "Efficient use of local edge histogram descriptor," *Proceedings of ACM International Workshop on Standards, Interoperability and Practices*, Marina del Rey, California, USA, 2000, pp. 52–54.

[21] S. J. Yoon, D. K. Park, C. S. Won and S.-J. Park, "Image retrieval using a novel relevance feedback for EHD of MPEG-7," *Proceedings of IEEE International Conference On Consumer Electronics (ICCE'01)*, Los Angeles, California, USA, 2001, pp. 354–355.

15

Shape Descriptors

Miroslaw Bober[1], F. Preteux[2] and Whoi-Yul Yura Kim[3]

[1]*Visual Information Laboratory, Mitsubishi Electric Information Technology Centre Europe, Surrey, United Kingdom,* [2]*Institut National des Télécommunications, Evry, France,* [3]*School of Electrical and Computer Engineering, Hanyang University, Seoul, South Korea*

15.1 INTRODUCTION

Object shape features are very powerful when used in similarity search and retrieval. This is because the shape of objects is usually strongly linked to object functionality and identity. Humans can recognize characteristic objects solely from their shapes – an illustration that shape often carries semantic information. This property distinguishes shape from some other elementary visual features, such as color or texture. A large body of research has been devoted to shape-based recognition, retrieval and indexing [1].

Many applications, including those concerned with visual objects retrieval or indexing, are likely to use shape features. Shape descriptors (SDs) can be used in e-commerce, where it is difficult to use a text index to specify the required shape of the object, and query-by-example (QBE) is simpler and faster. In fact, any application targeting visual scene interpretation and object recognition cannot do it efficiently without powerful shape tools.

This chapter is concerned with techniques for describing and matching shape features of 2-D and 3-D objects. The Visual Part of the MPEG-7 standard defines three descriptors with different properties: the region-based shape, the contour-based shape, and the 3-D Shape spectrum descriptors. The theory underpinning each of them, their respective properties and application domains are explained here in some detail. We also present examples of applications demonstrating the ability and applicability of the MPEG-7 SDs.

In the following section, an overview of the MPEG-7 shape descriptors is presented and their functionalities, properties and application domains are analyzed. Section 15.3 describes the design philosophy, performance evaluation and test sets used. The region-based SD, contour-based SD and 3-D SDs are presented in Sections 15.4, 15.5 and 15.6,

respectively. Section 15.5.6 deals with a combination of the multiview and contour-SDs to describe 3-D objects. Example applications are outlined in Section 15.7 and conclusions are drawn in Section 15.8.

15.2 OVERVIEW OF SHAPE DESCRIPTORS

The notion of object shape and shape similarity, while intuitively clear, may have many meanings. Most of the real-world objects are 3-D objects and a 3-D shape descriptor developed by MPEG [2, 3] is presented in Section 15.6. However, in the image and video world, 2-D projections of the real-world 3-D objects onto an image plane are visible and often their exact 3-D shapes are unknown. To cover this important domain, MPEG-7 provides *contour-based shape* and *region-based shape* tools.

Figure 15.1 shows that even in the 2-D case, there can be two notions of similarity based on the object contour and the distribution of all pixels within contour. Objects in the first row have similar spatial distribution of pixels and are therefore similar according to a region-based criterion. However, they clearly have different outline contours. When contour-based similarity is concerned, objects shown in each column are similar. Posing a query with the object located in the first row and second column will result in retrieved object from the first row (if region-based similarity is concerned) or second column (if contour-based similarity is concerned). MPEG-7 supports both notions of similarity having *region-based* and *contour-based SDs*.

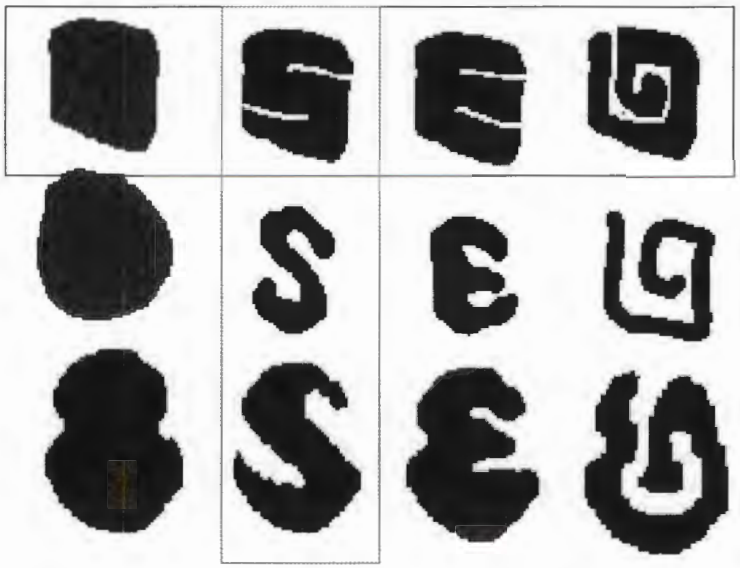


Figure 15.1 Example of contour-based and region-based region similarity (© 2001 IEEE, from M. Bober; MPEG-7 Visual Descriptors, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 6, June 2001)



Figure 15.2 Example of shapes in which region-based shape is applicable

15.2.1 Region-Based Shape Descriptor

The region-based SD expresses pixel distribution within a 2-D object or region. Since it is based on both boundary and internal pixels, it can describe complex objects consisting of multiple disconnected regions as well as simple objects with or without holes (Figure 15.2). The region-based SD belongs to the broad class of shape analysis techniques based on moments [19]. It uses a complex 2-D Angular Radial Transformation (ART) [2, 4] defined on a unit disk in polar coordinates. Some important features of this descriptor are:

- it gives a compact and efficient way of describing properties of multiple disjoint regions simultaneously;
- it can cope with errors in segmentation in which an object is split into disconnected subregions, provided that descriptor was computed using all subregions;
- the descriptor size is compact and fixed to 17.5 bytes; and
- the descriptor is robust to segmentation noise.

The ART descriptor is considered in detail in Section 15.4.

15.2.2 Contour-Based Shape Descriptor

The contour-SD is based on the Curvature Scale-Space (CSS) representation of the contour [2, 5]. Objects for which characteristic shape features are contained in their contour are described efficiently by the contour-SD – see Figure 15.3. Some important features of this descriptor are:

- it emulates well the shape similarity perception of the human visual system;
- it is robust to significant nonrigid deformations (e.g. the outline of a running person or walking beetle) and to perspective transformations; and
- it is compact and the descriptor size adjusts to the contour complexity (14 bytes on average).



Figure 15.3 Examples of shapes in which contour-based shape is applicable

Note that, as shown earlier, objects may have similar region-shape properties and still exhibit different contour properties (and vice versa). The descriptor, its properties and example applications are considered in detail in Section 15.5.

15.2.3 3-D Shape Descriptor

The 3-D shape spectrum descriptor expresses characteristic features of objects represented as discrete polygonal 3-D meshes. It is based on the histogram of local geometrical properties of the 3-D surfaces of the object. Some important features of this descriptor are:

- it can retrieve articulated objects with different postures;
- it is robust to scale and Euclidean transforms;
- it is robust to underlying meshed representation.

Section 15.6 describes the 3-D SD and its properties.

15.2.4 Multiple-View Descriptor for Shape

Sometimes a 3-D object model may not be known, but many applications require more than just 2-D information. Let's take e-shopping as an example – the customer may imagine the required product, say a flower vase, but there is no actual 3-D object from which a 3-D descriptor could be extracted. Such a mental picture consists of a set of 2-D views, which when combined describe the 3-D properties of the object. To cater for such situations, a combination of a *Multiple-View descriptor* and *Contour-shape* or *Region-SD*s can be used.

15.3 DESIGN AND EVALUATION OF THE SHAPE DESCRIPTORS

A good descriptor captures characteristic shape features in a concise manner, which should render itself easily to fast search and browsing. It is not required that the original shape need be reconstructed from the descriptor and therefore a significant compactness of the representation can be achieved. Naturally, it should be invariant to scaling, translation and rotation. Furthermore, as one is frequently dealing with digital (sampled) images, scaling of objects to a relatively small size (as compared with the sampling grid) may result in significant distortions in their shape. It is therefore critical that the shape representation should be robust to such effects. Other types of distortions often result from the imaging conditions, for example, a 2-D shape may undergo a perspective transformation as a result of change in the viewing angle. Some objects are flexible and the representation should be able to cope with nonrigid deformations. Finally, the descriptors should cope with distortions typical for object segmentation and extraction processes.

15.3.1 Evaluating Contour-Based Shape Descriptors

Performance evaluation of the contour-based descriptors was performed in four tests: robustness to rotation, robustness to scaling, similarity-based retrieval and robustness to nonrigid deformation. Since the notion of shape similarity is subjective and it may be

sometimes difficult to precisely rank similarity, a new way of assessing performance was developed. The idea is to construct the database in such a way that the shapes create 'natural' clusters of similar shapes. The similarity between shapes within each cluster is greater than the similarity of shapes between clusters.

This is illustrated in Figure 15.4, in which cluster A contains shapes depicting horses (side view), cluster B contains shapes of camels and cluster C shapes of cattle. While there is a different degree of variability of shapes in each cluster (see the position of tail or legs), the degree of similarity between objects within each class is greater than the similarity between two objects from two different classes. This property simplifies the assessment of the retrieval performance: a query with a shape from a cluster A should ideally return all remaining shapes from cluster A at top ranks before shapes from any other clusters are retrieved. The *Similarity* test set has 70 object classes, each class including 20 shapes (1400 shapes in total). It contains examples of natural objects with significant shape variability, such as beetles (Figure 15.5, top row); manually-drawn objects (second

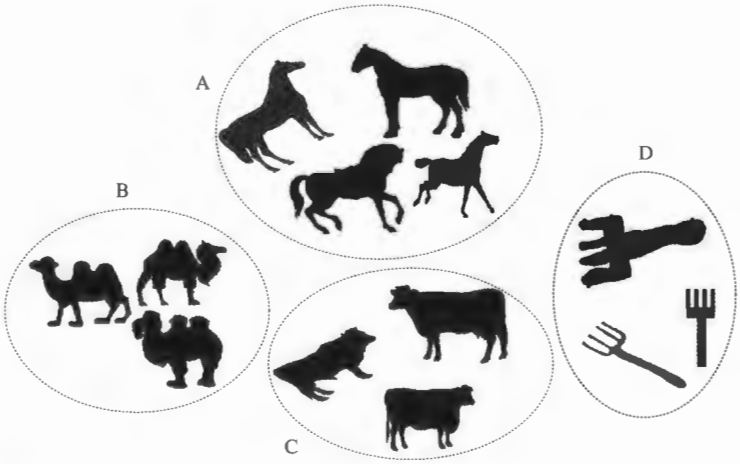


Figure 15.4 Example of shape classes in the contour-based shape test set

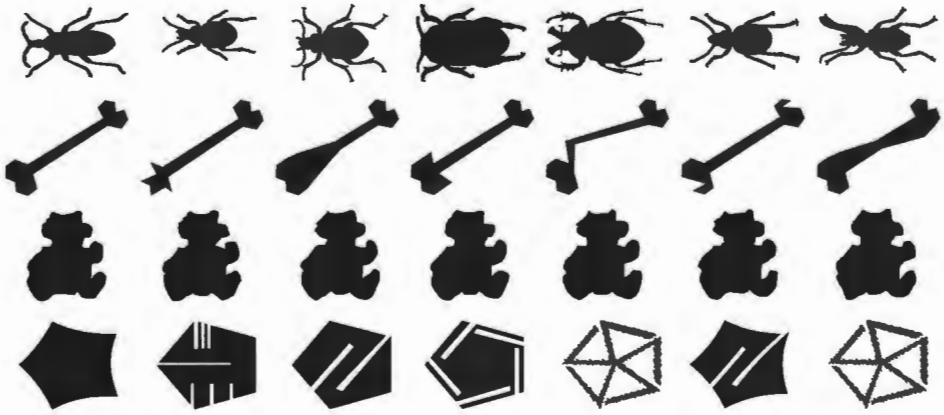


Figure 15.5 Example shape clusters from the database for similarity-based retrieval

row), objects extracted from cartoons with lesser amount of shape variability (third row) and drawn objects targeting specifically region-based similarity (bottom row). Each of the 1400 images is used as a query in turn and the number of correctly retrieved shapes in the top 40 is calculated. (An object is considered to be correctly retrieved if it comes from the same class as the shape used for query).

All the techniques tested supported rotational and scaling invariance, however, it was important to test the robustness of each approach to the effects such as changes to the shape due to image resampling (rotation & scaling), quantization of the representation and the overall behavior of the software implementations. For this reason, two additional databases were constructed to test scale invariance and rotation invariance. The *Scaling* database contains 70 reference shapes (one from each class) and the derived shapes, which were reduced (in each dimension) by factors: 0.3, 0.25, 0.2 and 0.1 and enlargement by a factor of two. The *Rotation* database contains versions of the 70 reference images digitally rotated by angles 9, 36, 45, 90 and 150 degrees.

The scale-invariance test is the more challenging of the two, because of the fact that some images are severely distorted when scaled by factors of 0.2 or 0.1. This was intentional – the objective is to determine the robustness to various degrees of distortions. Figure 15.6 shows examples in which scaling by a factor of 0.1 caused severe distortions, to a degree where even a human may find it difficult to determine a true shape.

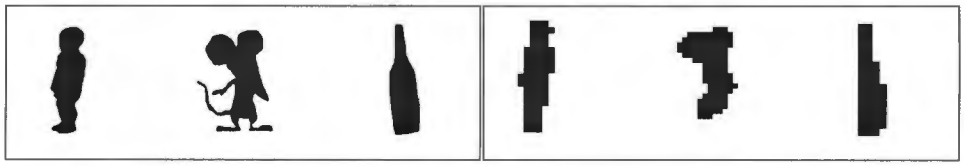


Figure 15.6 Shapes of child, rat and bottle (left box) and their scaled-down and resampled versions (right box)

The main characteristic features of object shape are usually retained under some nonrigid deformations. Many applications will rely on this property and therefore it is important that good SDs can still cope with such deformations. The *Nonrigid* test set contains 1300 shapes, including 200 frames showing a swimming fish (often known in MPEG community as the Bream fish) and 1100 unrelated shapes of various marine creatures, including many images of other fish. The shape of the Bream fish is deformed as a result of swimming and also the fish turns causing more dramatic changes to its shape. The query is posed with the first image of the Bream fish and the number of instances of other Bream fish in the top 200 retrievals is computed. Figure 15.7 shows the query shape (i.e. Bream fish – left box), three shapes of the Bream fish under nonrigid deformations (middle box) and some other fish present in the database (right box).



Figure 15.7 Example of shapes used to test robustness to nonrigid deformations

15.3.2 Testing Region-Based Shape Descriptors

Region-based techniques were tested in a separate experiment. A database of trademarks is used to evaluate performance on complex shapes consisting of multiple disjoint regions, possibly with holes. The *Classified* test set contains 2800 trademark shapes: 678 objects are classified into 10 classes, on the basis of perceived region-shape similarity (variable number of shapes in each class) and the 2122 trademarks remain unclustered. Examples of similar shapes in two classes are shown in Figure 15.8.

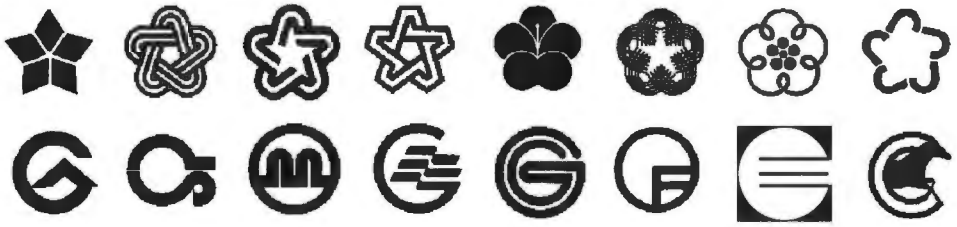


Figure 15.8 Example of shape classes in the *Classified* region-based shape test set

The *Digital-scale* and *Digital-rotation* test sets are based on 20 randomly selected trademarks that were scaled by factors of 3, 1.5, 0.5 and 0.33 and rotated by 9, 36, 45, 90, 120 and 150 degrees.

The *Sensor-rotation&scale* test set was obtained by capturing a video sequence with 20 different trademarks printed on a board and subsequently digitizing certain images from the sequences. Scaling and rotation is obtained by changing the camera-to-object distance and rotating the board with the shape. Both, analog and digital cameras were used.

Finally, the *Perspective* data set was created by capturing video sequences containing 20 different reference trademarks and selecting for each reference shape 10 images exhibiting different degree of perspective distortions (Figure 15.9). All transformed data sets are combined with the database of 2800 images as the background images.



Figure 15.9 Examples of shapes under perspective transformation used in tests

15.3.3 Testing 3-D Shape Descriptors

Constructing a database for testing 3-D descriptors was perhaps the most difficult task due to lack of a sufficiently large publicly available data set. Some 1290 models were collected by combining the 3-D library used for MPEG-4 experiments, 3-D café data set (*http://www.3dcafe.com*) and models of letters generated specifically for testing. The data set was divided into 15 broad categories, such as *4-limbs*, *cars*, *aerodynamic-shapes*, *trees, balloons, letter-a, letter-b* and so on. As in the previous experiments all classified

models were used as queries and the Bull's Eye score was used (Bull's-Eye Percentage (BEP)). The BEP is defined for each query as the percentage of correct matches (i.e. items within the same category as the query) among the 2Q top retrieved results, in which Q denotes the size of the considered query category.

In the following sections we will introduce each SD, namely, the region-based SD, the contour-based SD and the 3-D SD.

15.4 REGION-BASED SHAPE DESCRIPTOR

In this section, we present the region-based SD. This descriptor takes into account all pixels constituting the shape, that is both the boundary and interior pixels. It is therefore applicable to objects consisting of a single connected region or multiple regions, possibly with holes. Conceptually, the descriptor works by 'decomposing' the shape into a number of orthogonal 2-D basis functions (complex-valued), defined by the Angular Radial Transform (ART). The normalized and quantized magnitudes of coefficients are used to describe the shape. The ART transform is defined in the next section, followed by the definition of the region-SD.

15.4.1 ART Transform

ART is the orthogonal unitary transform defined on a unit disk that consists of the complete orthonormal sinusoidal basis functions in polar coordinates. The ART coefficients are defined by:

$$F_{nm} = \langle V_{nm}(\rho,\theta), f(\rho,\theta)\rangle = \int_0^{2\pi}\int_0^1 V^*_{nm}(\rho,\theta)\, f(\rho,\theta)\, \rho \,\mathrm{d}\rho\,\mathrm{d}\theta \tag{1}$$

where F_{nm} is an ART coefficient of order n and m, $f(\rho,\theta)$ is an image function in polar coordinates and $V_{nm}(\rho,\theta)$ is the ART basis function that are separable along the angular and radial directions, that is,

$$V_{nm}(\rho,\theta) = A_m(\theta)R_n(\rho) \tag{2}$$

In order to achieve rotation invariance, an exponential function is used for the angular basis function,

$$A_m(\theta) = \frac{1}{2\pi}\exp(jm\theta) \tag{3}$$

The radial basis function is defined by a cosine function,

$$R_n(\rho) = \begin{cases} 1 & n = 0 \\ 2\cos(\pi n\rho) & n \neq 0 \end{cases} \tag{4}$$

Real parts and imaginary parts of ART basis functions are shown in Figure 15.10.

15.4.2 Descriptor Representation

The ART descriptor is defined as a set of normalized magnitudes of complex ART coefficients. Rotational invariance is obtained by using the magnitude of the coefficients.

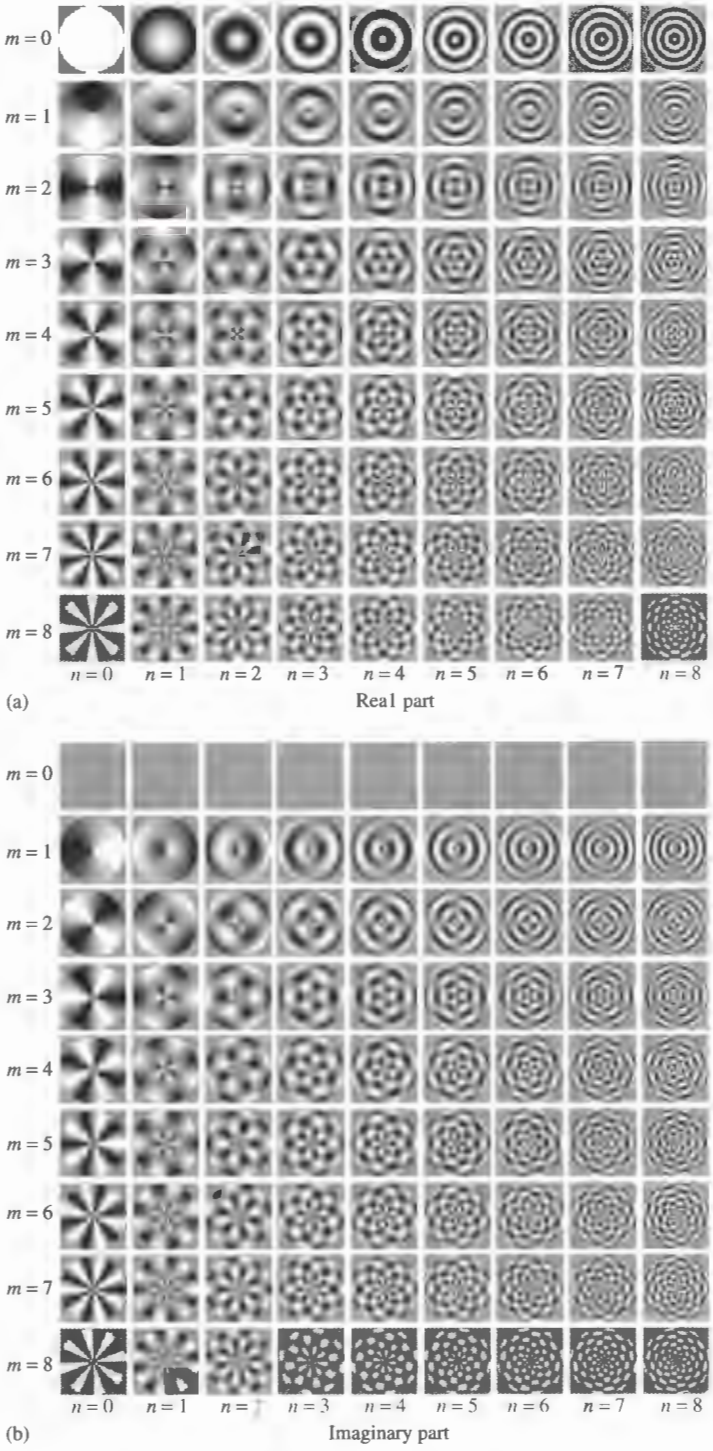


Figure 15.10 Real and imaginary parts of ART basis functions

Table 15.1 Representation of the region-SD

Field	Number of bits	Meaning
Magnitude of ART[k]	4	An array of 35 normalized and quantized magnitudes of the shape coefficients.

Twelve angular and three radial functions are used ($n < 3, m < 12$). For scale normalization, ART coefficients are divided by the magnitude of ART coefficient of order $n = 0$, $m = 0$. The ART coefficient of order $n = 0$, $m = 0$ is not used as a descriptor element because it is constant after scale normalization. To keep the descriptor size to a minimum, quantization is applied to each coefficient using four bits per coefficient. Table 15.1 shows the representation of the ART descriptor.

15.4.3 Similarity Measure

The distance (or dissimilarity) between two shapes described by the ART descriptor is calculated using an L-1 norm, for example, by summing up the absolute differences between ART coefficients of equivalent order.

$$\text{Dissimilarity} = \sum_i \|M_d[i] - M_q[i]\| \tag{10}$$

Here, the subscript d and q represent image in the database and query image, respectively and M is the array of ART descriptor values.

15.4.4 Experimental Results

Table 15.2 presents the results obtained in the experiments using the ART descriptor implementation by MPEG. Good results are obtained in tests of the rotational and scaling invariance. The performance is somewhat degraded under a perspective transformation test. Finally, the tests of similarity-based retrieval (Classified test set) show a score of more than 66%, which is high considering the complexity of the test database.

Figure 15.11 shows an example of the retrieval results, using a query image for which the performance is close to the average obtained in *Similarity* retrieval test. There are 22 shapes in this cluster. According to the manual clustering, shapes numbered 4, 5,

Table 15.2 Experimental results for the region SD on the trademark database

Test set	Digital-scale	Digital-rotation	Sensor rotation & scale	Perspective	Classified	Overall[a]
ART-eXperimentation Model (XM)	86.20%	98.06%	98.76%	62.65%	66.13%	76.28%

[a] Averate performance = {(Digitalscale + DigitalRotation + SensorRotation & scale + Perspective)/4 + Classified}/2.



Figure 15.11 Example similarity retrieval results for a trademark database

Table 15.3 Experimental results for the region-SD on the contour-shape data sets

Test set	Scaling	Rotation	Similarity	Nonrigid	Overall[b]
ART-XM	97.60%	100%	68.34%	92%	86.40%

[b]Overall result = {(Scaling + Rotation)/2 + Similarity + Non-rigid}3.

7, 8, 9, 13 and 16 are considered members of a different cluster (therefore counted as errors), however, a large degree of similarity is still visible between them and the query image. Note that only region-SD can be used for such complex shapes that cannot be expressed by a single contour.

Experimental results on the *Contour-shape test sets* are presented in Table 15.3. It can be seen from *Rotation* and *Scaling* tests that the descriptor offers very good rotational and scaling invariance. As mentioned earlier, the region-based approach is more robust to deformations related to the extreme scaling – in this part it outperforms the contour-SD by 7%. In *Similarity*-based retrieval, the region-SD is on average about 15% worse compared with the contour-SD. Details on experiments using contour-shape test sets are provided in Section 15.5.

The experiments confirm that the main strength of the region-based shape techniques is that they perform well for shapes where region-based similarity is important, for example, complex shapes that consist of several disjoint regions, such as trademarks or logos, emblems, clip art and characters. The performance under perspective transformation can be improved from 62.65% to 97.69% when combined with a shape-normalization technique called a compaction algorithm while preserving performance in other tests [6].

15.5 CONTOUR-SHAPE DESCRIPTOR

In this section, the contour-SD is presented: we explain the underlying theory, discuss its properties and overview some experimental results. The contour-SD is based on the CSS representation of the contour [5, 7]. The CSS representation has been successfully used for search and retrieval in the past, but it has been further extended and optimized during the MPEG development phase. Six different techniques were compared in MPEG experiments: Fourier-based techniques, shape representations based on Zernike moments and ART, turning angles and wavelet-based techniques. It was concluded that the CSS based descriptor offered the best overall performance for contour-based similarity. A detailed overview of the CSS representation and its use in the MPEG-7 standard is presented in [7].

15.5.1 The CSS Representation

The concept of CSS representation is based on an observation that when comparing shapes, humans tend to decompose shape contours into concave and convex sections. The overall similarity is assessed on the basis of similarity of the 'corresponding' sections: for example, how prominent they are, their length relative to the contour length and their position and order on the contour. The CSS representation also 'decomposes' the contour into convex and concave sections by determining the inflection points (e.g. points at which curvature is zero). This is done in a multiresolution fashion, where the contour is analyzed at various scales, obtained by a smoothing process. Graphically, this can be illustrated by a so-called CSS image, which can be constructed for any contour (open or closed). Basically, the CSS image shows how the inflection points change when filtering is applied to the contour: CSS x-axis corresponds to the position on the contour (clockwise, starting from any arbitrary point) and y-axis corresponds to the values of a shape smoothing parameter (when y-values increase, amount of smoothing increases). Any black point in the CSS image signifies that at the corresponding position on the contour and at the corresponding scale, there is an inflection point.

Figure 15.12 shows shape of a fish (left) and its CSS image (right). There are three peaks in the CSS image, (marked as A, B and C) signifying that the fish contour has three concave parts (marked correspondingly on the contour). The CSS image also shows that the concave segments A and C are approximately of similar perceptual importance (because the peak heights are similar), while concavity B is less prominent. We can also deduce that concavity B is located in-between concavities A and C and find the relative distances between the inflection points.

Extraction of the CSS representation involves calculation of the curvature of a (usually digital) contour, while progressively smoothing the contour. Figure 15.13 shows the shape evolution during filtering (left column), the corresponding curvature functions (middle column) and the CSS image (right column). The curvature of a contour is defined as the derivative of the tangent angle to the contour. Let us assume that the contour C is represented by a parametric vector equation: $C(u) = [x(u), y(u)]$, where x, y are the coordinates of the pixels belonging to the contour and u is a parameter. For a closed, planar contour, with u as the normalized arc length parameter, the curvature can be calculated as:

$$\kappa(u) = \dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u) \tag{12}$$

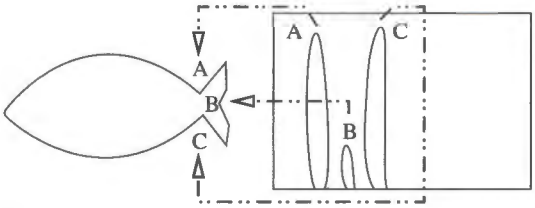


Figure 15.12 Example of the CSS image representation for a fish contour

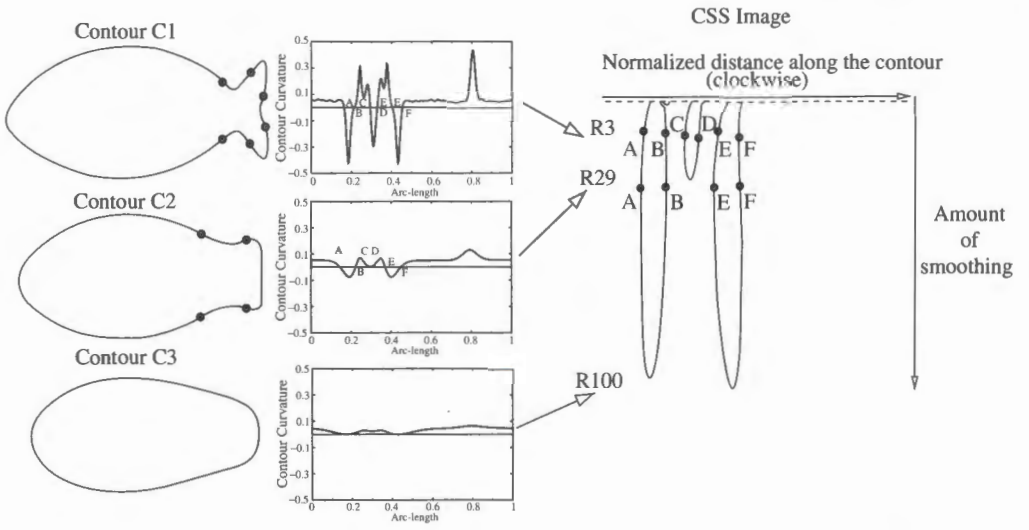


Figure 15.13 Extraction of the CSS image for the fish contour of Figure 15.12

where $\dot{x}(u), \ddot{x}(u)$ are the first- and second-order derivatives of the x with respect to the parameter u (and similarly $\dot{y}(u), \ddot{y}(u)$). The smoothing is performed iteratively and for each realization of the contour (e.g. after each iteration), the zero crossings of the curvature function are computed. The filtering process continues until a convex contour results, that is, until there are no more zero crossings of the curvature function. Figure 15.13 shows three contours C1, C2, C3 after increasing amount of smoothing has been applied and their corresponding curvature functions (middle column). The zero crossings of the curvature function are marked on each contour. Contour C3 is convex and therefore there are no zero-crossings in its curvature function. Each zero-crossing has a 2D vector associated with it (s,l), where s is the amount of the smoothing applied and l the position of the zero-crossing on the contour. l is expressed as the distance along the contour from an arbitrarily chosen point on the contour to the inflection point, while circulating the contour clockwise. The CSS image is obtained by plotting all the zero-crossing points on a 2-D plane, where vertical axis corresponds to the amount of filtering and horizontal axis corresponds to the position on the contour. Thus, each contour C1, C2, C3 defines a single horizontal line in the CSS image, shown in Figure 15.13 (right).

15.5.2 Descriptor Representation and Extraction

The descriptor consists of the eccentricity and circularity values of the original and filtered contour (each quantized to 6 bits), the index indicating the number of peaks in the CSS image (6 bits), the magnitude (height) of the largest peak (7 bits) and the x- and y-positions on the remaining peaks (9 bits per peak). The average size of the descriptor is 112 bits per contour. The table below summarizes the binary syntax of the contour-SD.

Field	Number of bits	Meaning
Num of Peaks	6	Number of peaks in the CSS image
GlobalCurvature	2*6	Circularity and eccentricity of the contour
PrototypeCurvature	2*6	Circularty and eccentricity of the smoothed (convex) contour
HighestPeakY	7	Absolute height of the highest peak (quantized)
peakX[]	6	X-position on the contour of a peak (quantized, relative to the highest, clockwise)
peakY[]	3	Height of the peak (relative to the previous peak's heigt and quantized)

Extraction of the circularity and eccentricity

The *circularity* of the contour is calculated using Eq. (13) below. It is uniformly quantized to 6 bits in the range [12–110). If the *circularity* value calculated is above 110, the value is clipped to 110.

$$\text{circularity} = \frac{\text{perimeter}^2}{\text{area}} \tag{13}$$

The calculation of *eccentricity* is as follows:

$$\text{eccentricity} = \sqrt{\frac{i_{20} + i_{02} + \sqrt{i_{20}^2 + i_{02}^2 - 2i_{20}i_{02} + 4i_{11}^2}}{i_{20} + i_{02} - \sqrt{i_{20}^2 + i_{02}^2 - 2i_{20}i_{02} + 4i_{11}^2}}} \tag{14}$$

where: $i_{02} = \sum_{k=1}^{M} (y_k - y_c)^2$, $i_{11} = \sum_{k=1}^{M} (x_k - x_c)(y_k - y_c)$, $i_{20} = \sum_{k=1}^{M} (x_k - x_c)^2$, M is the number of points inside the contour shape and (x_c, y_c) is the center of mass of the shape. Eccentricity is uniformly quantized to 6 bits in the range [1–10). If the *eccentricity* value calculated is above 10, the value is clipped to 10.

Extraction of the peak parameters

The CSS image does not have to be explicitly extracted, we only need the position of the peaks. Fast methods of extraction are described in [2, 7]. Here we outline a simple but slower procedure for ease of understanding. Firstly, N equidistant points are selected on the contour, starting from an arbitrary point and following the contour clockwise. The x-coordinates and y-coordinates of the selected N points are grouped into two series

X, Y. The smoothing of the contour is performed by repetitive application of a low-pass filter with the kernel (0.25, 0.5, 0.25) to X- and Y-coordinates of the selected N contour points. As a result of the smoothing, the contour evolves and its concave parts gradually flatten out, until the contour becomes convex (no zero crossings in the curvature), at which point the iterations terminate. For each smoothed contour, the zero crossings of its curvature function are computed. The CSS image is constructed by marking each zero-crossing at an appropriate position. The CSS image horizontal-coordinates correspond to the indices of the contour points selected to represent the contour $(1, \ldots, N)$ and CSS image vertical-coordinates correspond to the amount of filtering applied, defined as the number of passes of the filter. So if the contour obtained after $k = 20$ iterations has a curvature zero crossing at location corresponding to index $I = 60$, the pixel (60,20) will be black. Once all zero crossings are marked, the locations of the prominent peaks (x_css, y_css) in the CSS image are extracted. The location values are suitably normalized so that they are invariant to the number of sampling points and the size of the contours. The peaks are ordered on the basis of decreasing values y_css, transformed using a nonlinear transformation and quantized. The details of the quantization are explained in [2].

15.5.3 Properties of the Contour-Shape Descriptor

The contour-SD has some very useful properties. Firstly, the representation is invariant to orientation, scale and mirroring of the object contour. It has also been shown to be robust to noise in the contour [2]. Secondly, it retains the local information of the input shape (e.g. the order and neighboring information about the features of the shape). This is because every contour concavity and convexity has its own corresponding contour on the CSS image. Local information plays an important role in retrieval, for example, one can determine the number of concave and convex elements on the contour just by counting the number of the so-called peaks (maxima) in the CSS image.

The descriptor is also very efficient in applications in which high variability in the shape is expected owing to, for example, deformations in the object (rigid or nonrigid) or perspective deformations. It supports search for shapes that are semantically similar for humans, even when significant intraclass variability exists, such as in the images of *Beetles* shown in Figure 15.5. Finally, many objects have distinguishing features in properties of their contour while having similar region properties and the contour-SD can efficiently differentiate between such shapes. All the above properties can be clearly seen when analyzing the experimental results, described in the following section.

15.5.4 Experimental Results

The results obtained in the experiments are summarized in Table 15.4. It can be seen that very good results are obtained in the *Rotation* and *Scaling* invariance tests. However,

Table 15.4 Experimental results for the contour-SD

Test set	Scaling	Rotation	Similarity	Nonrigid	Overall
CSS-XM	91.03%	100%	79.15%	96%	90.22%

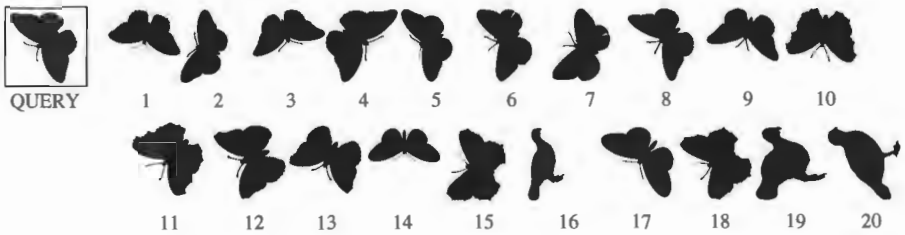


Figure 15.14 Example similarity-based retrieval results for the contour-SD

the real strength of the contour-SD is in its performance in similarity-based retrieval (*Similarity*), where the average result is above 79%.

Figure 15.14 shows example results for a similarity-based retrieval, for the butterfly-1 query image. It can be seen that the majority of butterfly images are retrieved correctly despite the significant variability in the shape of the wings, the size and spread of the antennae and whether or not the corpus of the body is protruding at the back. Images 16, 19 and 20 are errors, but still some degree of similarity exists.

15.5.5 Region-based vs Contour-based Shape Descriptors

A class-by-class analysis of performance of the SDs in similarity-based retrieval illustrates very well the differences between the region-based and contour-based notion of similarity. Let us consider the shape classes in which relative performance of one of the descriptors is better by 50% or more, compared with the other descriptor. This means that one of the descriptors retrieved 50% more correct shapes for the given class, which is a very significant difference in performance.

The contour-SD outperforms the region shape by 50% or more for 17 classes – Figure 15.15 shows example shapes.

The first class includes images of a running person, where significant nonrigid deformations occur. The contour-based technique is much more robust to such deformations. *Fork* (second object in Figure 15.15) is a good example of a shape in which

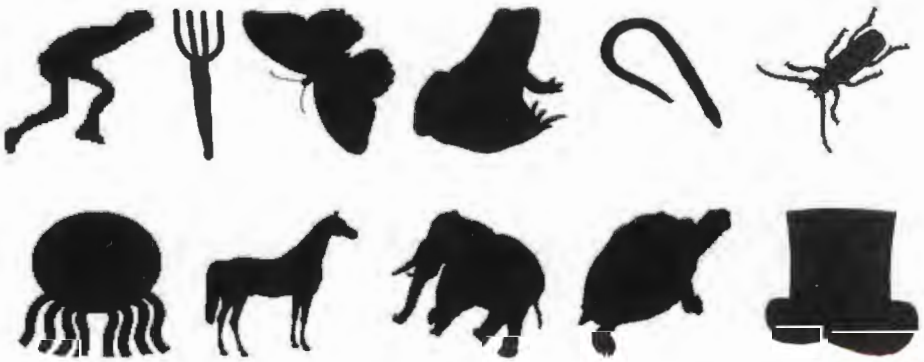


Figure 15.15 Example of shapes in which contour descriptor significantly outperforms region descriptor



Figure 15.16 Classes in which the region-SD significantly outperforms the contour-SD

characteristic features are conveyed in the contour. The region-based approach tends to confuse it with other elongated objects that expand at one end (e.g. *Spoon*, *Hammer*). Similarly, for *Butterfly* some characteristic features are in its antennae, which have a relatively small area and thus are not sufficiently emphasized by the region-based approach. Finally, the classes of *Beetles, Horses* and *Elephants* are examples in which the natural variability in shapes is much better captured by the contour-based approach.

The four classes in which the region-based approach performs better are shown in Figure 15.16. The first, second and fourth shapes represent classes in which the same 'basic convex shape' such as *Circle* (*Device-9*), *Triangle* (*Device-4*) and *Pentagon* (*Device-6*), has been altered by cutting out a variable number of long and narrow openings (Refer to Figure 15.5, bottom row, to see some shapes in the 'pentagon' class). All such shapes have a similar distribution of pixels within their regions, but obviously very different contours, because the number and shape of the openings differs within each class. Finally, the third shape, a square with rounded edges, is a convex shape and therefore its contour carries very little characteristic information (no peaks in the CSS image). The contour-based technique tends to confuse it with other convex objects, for example, with the *Circle*.

15.5.6 Combining a Multiple-View and 2-D Shape Descriptors

The Multiple-View Descriptor [2] has been developed to combine 2-D descriptors representing a visual feature of a 3-D object seen from different view angles [8]. The descriptor forms a complete 3-D view-based representation of the object. Any 2-D visual descriptor, such as, for example, contour-shape, region-shape, colour or texture can be used. The Multiple-View Descriptor supports integration of the 2-D descriptors used in the image plane to describe features of the 3-D (real-world) objects. Experiments with the Multiple-View Descriptor and the Contour-based SD have shown good performance in multi-view description of 3-D shapes [8].

15.6 3-D SHAPE DESCRIPTOR

Intuitively, the shape concept of a 3-D object is intrinsically independent of its spatial location, size and meshed representation. Hence, a 3-D SD has to be invariant to geometric (Euclidean and scaling) transforms and robust to the multiple topological representations (different triangulations) of the mesh.

Most of the time, a semantic meaning is implicitly associated with the shape concept. Let us exemplify this purpose by considering an articulated object like a *humanoid*.

The generic word *humanoid* recovers *de facto* a large family of *geometric shapes* that can highly differ. In addition, an element of this family may have several appearances, since a humanoid is an *articulated* and deformable *object.* A sitting man and a standing-up man are both humanoids. Are their associated geometric shapes similar? To answer this question properly it requires to specify if the relative positions of each component of the articulated object are or are not taken into account. Humans have the ability to decide when such an information is relevant or not, depending on specific purposes. Recognizing humans, regardless of their postures, does not need any additional information although gesture recognition, however, requires that the relative position information of the object components is available. Computers, however, do not have any mechanism for deciding. Consequently, the semantic meaning associated with the shape concept should be specified so that the computer-based system performs the appropriate technical solutions, for example, local SD for discriminating between articulated objects that may have different postures and global SD for gesture recognition applications.

Considered within the framework of shape-based retrieval applications, SD should have reasonable size and should support simple similarity measures, enabling effective search or browsing of possibly very large databases. To fulfil such practical constraints, global-shape descriptions were proposed in the past.

The above-considerations led to state the following requirements to be fulfilled by a 3-D SD. First, a 3-D SD should locally characterize free-form surfaces represented as discrete polygonal 3-D meshes. A surface-based approach is motivated for the sake of generality, as 3-D meshes may include open surfaces that do not have any associated volume. Then, the 3-D SD should be invariant with respect to scale and Euclidean transforms and robust to different triangulations of the same object. Finally, a 3-D SD should have the ability to successfully retrieve articulated objects with different postures. The 3-D SSD, introduced in the following section, fulfils well all these requirements.

15.6.1 The 3-D Shape Spectrum Descriptor

The principle of the 3-D SSD consists of expressing local geometrical attributes of the 3-D surfaces. More precisely, the 3-D SSD of a 3-D mesh is defined as the histogram of the shape index values, calculated over the entire mesh.

The shape index, first introduced by Koenderink [9], is defined as a function of the two principal curvatures. Let p be a point on a regular [10] 3-D surface S and let k_p^1 and k_p^2 denote the principal curvatures [1, 10] associated with point p. The *shape index* at point p, denoted by I_p, is defined as:

$$I_p = \frac{1}{2} - \frac{1}{\pi} \operatorname{arctg} \frac{k_p^1 + k_p^2}{k_p^1 - k_p^2} \qquad \text{where } k_p^1 \geq k_p^2 \tag{15}$$

The shape index, is a local geometrical attribute of a 3-D surface, expressed as the angular coordinate of a polar representation of the principal curvature vector. The shape-index ranges in the interval [0,1) and is not defined for planar surfaces. The shape index provides a scale for representing salient elementary shapes such as convex, concave, rut, ridge and saddle (Figure 15.17) and is invariant with respect to scale and Euclidean transforms.

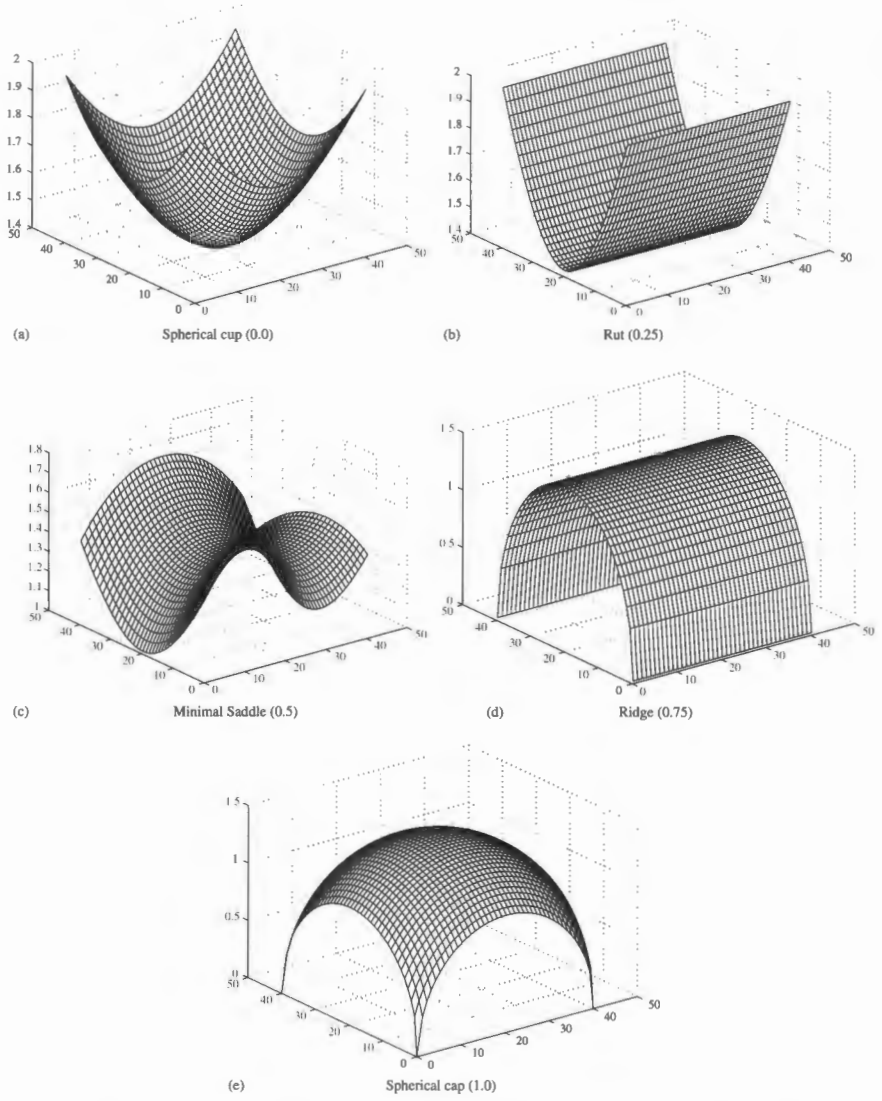


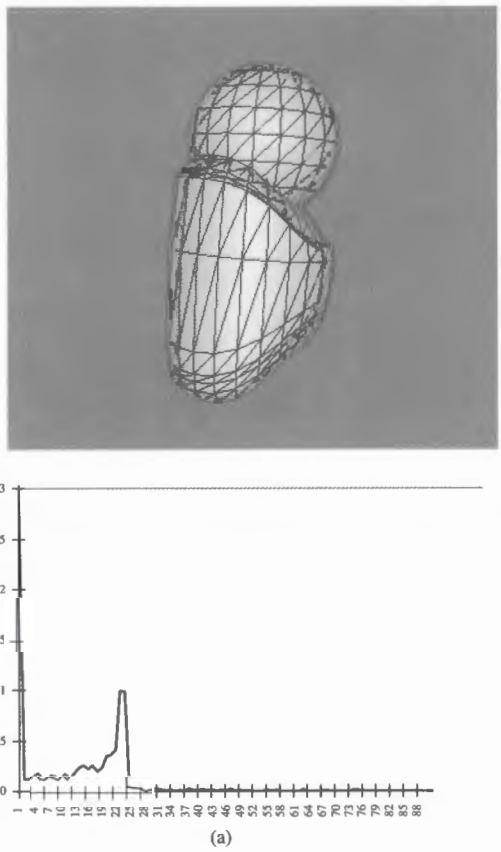
Figure 15.17 Elementary shapes and their corresponding shape index (in brackets)

The 3-D SSD is defined as the distribution of the shape index over the entire mesh and is calculated as follows. The range of the 3-D shape index values, namely, the [0, 1) interval, is uniformly divided into a set of N_{bins} intervals, $\{\Delta_k\}_{k=1}^{N_{\text{bins}}}$, where

$$\Delta_k = \left[\frac{k-1}{N_{\text{bins}}}, \frac{k}{N_{\text{bins}}}\right], \quad \text{for } k = \{1, 2, \ldots, N_{\text{bins}}\} \text{ and } \Delta_{N_{\text{bins}}} = \left[\frac{N_{\text{bins}}-1}{N_{\text{bins}}}, 1\right] \tag{16}$$

The relative (with respect to the total area of the mesh) area of each face of the mesh of shape index belonging to the interval Δ_k is added to the kth component of the 3-D SSD.

Two additional components, denoted by *PlanarSurface* and *SingularSurfaces* and representing the relative surfaces of planar and border patches, respectively, are introduced in the 3-D SSD. The planar surfaces must be separately considered since the 3-D shape index is not defined for planar patches. Let k_a denote the l_2 norm of the curvature vector $(k_1, k_2) : k_a = \sqrt{k_1^2 + k_2^2}$. We define the degree of curvedness of the surface as $C = \text{Area}\{F_0(f_i)\} * k_a$, where $F_0(f_i)$ denotes the set i of zero-adjacent faces[1] of face f_i. Each face with curvedness C lower than a threshold T, is considered as part of a planar patch and its relative area is used to increment the *PlanarSurface* component of the 3-D SSD. Unreliable estimates of the principal curvatures may occur in the case of border faces[2] with a number of zero-order adjacent faces smaller than a predefined number $N_{\min}$. Hence, the border faces are not taken into account when computing the shape index and their relative area is stored into the *SingularSurfaces* component. Thus, the 3-D SSD is a vector with $N_{\text{bin}} + 2$ components.

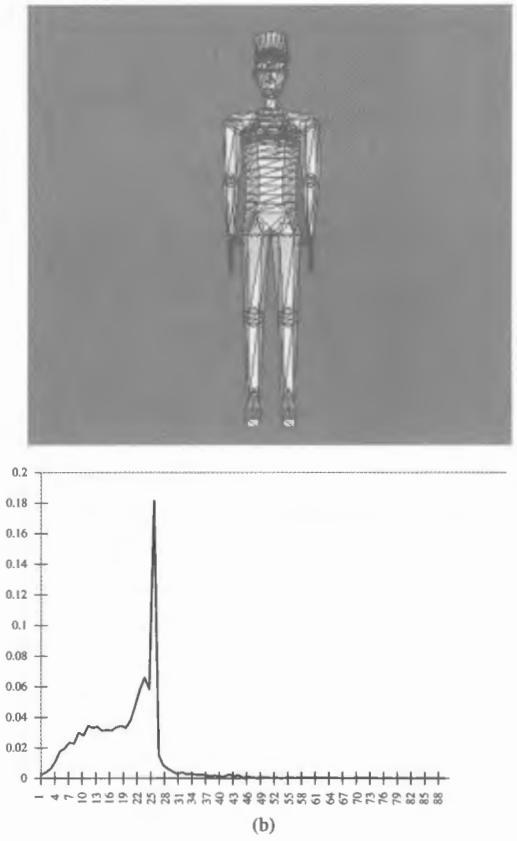


(a)

Figure 15.18 3-D mesh objects and their corresponding 3-D SSD's

[1] Two faces are zero-adjacent if and only if they have at least a common vertex.

[2] *Border faces* are defined as faces including at least a border edge. An edge belonging to a single face of the mesh is called a *border edge*.



(b)

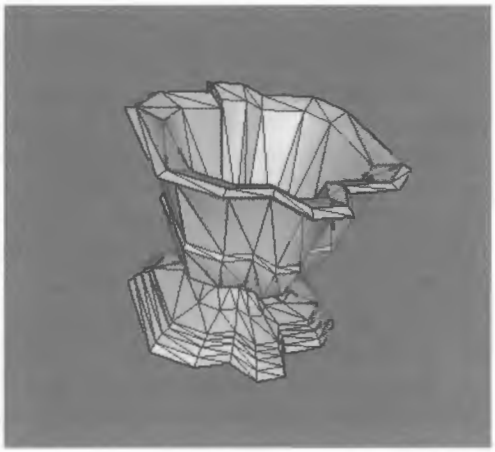
Figure 15.18 (*continued*)

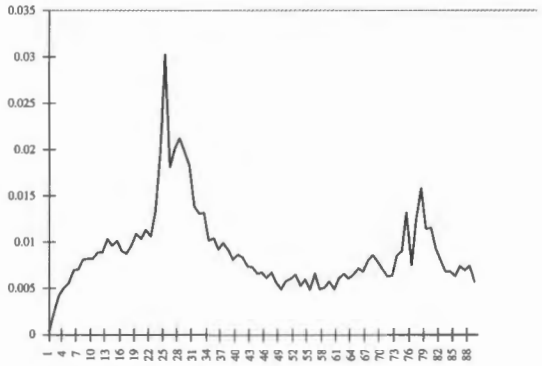
The mesh in Figure 15.18(a) is mainly composed of spherical and cylindrical surfaces. Consequently, the shape spectrum distribution has two peaks, corresponding to shape index values of 0 and 0.25. The humanoid in Figure 15.18(b) is mainly composed of piecewise cylindrical surfaces. Its associated 3-D SSD has a predominant peak around the shape index value of 0.25. Finally, the urn in Figure 15.18(c). is a surface with both interior and exterior components. Consequently, convex and concave surface patches are almost equally represented within a bimodal 3-D SSD with peaks around shape-index values of 0.25 and 0.75.

Let us note that the 3-D SSD is intrinsically invariant to geometric transforms. The 3-D SSD has been used for indexing and retrieving 2-D images [11] and 3-D range data [12].

15.6.2 Syntax and Semantics of the 3-D SSD

The 3-D SSD descriptor consists of the following elements: (1) *numberOfBins* – specifying number of bins used for the shape representation; (2) *bitsPerBin* – the number of bits used to represent descriptor values; (3) *Spectrum* – the values of the 3-D shape spectrum, where Spectrum[k] contains the relative area of all the 3-D mesh-surface regions of shape





(c)

Figure 15.18 (*continued*)

index lying in the interval [k/numberOfBins, (k + 1)/numberOfBins) quantized uniformly to bitsPerBin bits; (4) *PlanarSurfaces* – the relative area of planar surface regions of the mesh, with respect to the entire area of the 3-D mesh, which is expressed as a ratio in the range between 0 and 1 with uniform quantization to bitsPerBin bits; and (5) *SingularSurfaces* – expressing relative area of all the singular polygonal components, with respect to the entire area of the 3-D mesh.

The following Binary representation syntax is used

Field	Number of bits	Meaning
NumberOfBins	8	Number of bins in the spectrum representation
BitsPerBin	4	Number of bits per bin
Spectrum[]	1–12	Spectrum values (quantized)
PlanarSurfaces	1–12	the relative area of planar surface regions of the mesh, with respect to the entire area of the 3-D mesh
SingularSurfaces	1–12	Relative area of all the singular polygonal components, with respect to the entire area of the 3-D mesh

15.6.3 Computation of the 3-D Spectrum Shape Descriptor

Since the 3-D SSD of a mesh is defined as the distribution of the shape index, computing the 3-D SSD requires estimation of the index shape. The latter is based on the principal curvatures, the estimation of which is the key step in the 3-D SSD extraction. A detailed description of the principal curvature estimation can be found in the Annex on the DVD. Note that the estimation procedure is very simple and accurate. However, it is relevant only when dealing with orientable- and regular-enough meshes.

Orientable and smooth 3-D mesh

In order to illustrate the concept of orientable mesh, let us consider the example in Figure 15.19. With each triangle of the mesh, we associate a normal vector with a direction given by the traversal order of the triangle vertices (Figure 15.19a). A mesh is orientable if each two E-neighbor faces[3] have consistent orientations (Figure 15.19b), that is, they are traversed in opposite directions. Figure 15.19c shows a *nonorientable* mesh: the common edge (v_1, v_2) is traversed in the same direction in the two neighboring triangles. If a mesh has at least two E-neighbor faces with different orientations, the mesh is said to be nonorientable. We note that an edge of an orientable mesh will belong to at most two faces. Only orientable meshes need be considered when computing the shape index and the principal curvatures.

This orientability constraint is, however, not sufficient because estimating second-order derivatives only makes sense when dealing with surfaces that are regular and smooth enough. For example, in the case of a 3-D model irregularly sampled, the face area and mean-elongation-ratio distributions may widely vary (Figure 15.20a). In addition, such a 3-D mesh shows sharp features, violating the smoothness hypothesis necessary when considering surface derivatives.

In order to obtain smoother mesh representations, a preprocessing step consisting of a mesh subdivision algorithm based on Loop's subdivision scheme is applied [13]. The subdivision algorithm consists of two successive steps: (1) mesh resampling by midedge vertex insertion (Figure 15.21) and (2) low-pass filtering of the vertex coordinates. By choosing appropriate filter coefficients, the limit surface can be shown to be tangent plane smooth. In addition, such a subdivision scheme has the advantage of drastically

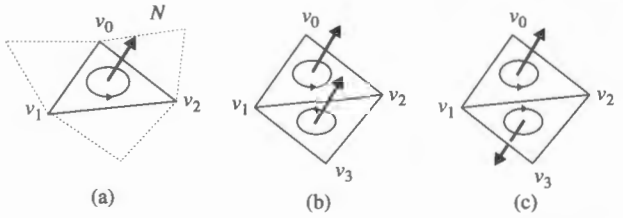


Figure 15.19 Principle of orientable mesh: (a) face orientation; (b) orientable mesh and (c) non-orientable mesh

[3] Two faces are said to be *E-neighbor faces* if they share at least a common edge.

The annex is available on the DVD as an electronic document.



(a) (b) (c)

Figure 15.20 (a) Initial model 'A'; (b) its refined versions after one; and (c) two subdivision levels

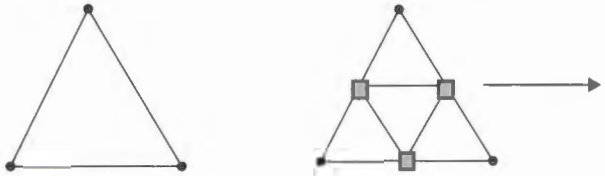


Figure 15.21 Mesh resampling by midedge insertion

reducing the relative area of the border faces. Let us note that such a schema requires a preliminary triangulation of the mesh models if they are made of arbitrary polygonal faces.

From now on, all the 3-D mesh to be indexed with the 3-D SSD are assumed to be orientable and smooth. But what about the 3-D SSD behavior with respect to multiple topological mesh representations?

Multiple topological representations and 3-D mesh prefiltering

VRML (Virtual Reality Modelling Language) data are initially intended for graphics purposes. Hence, 3-D objects can be meshed in many different ways resulting in totally different topological representations including possible degenerated or overlapped polygons. Let us briefly analyze the impact of such multiple topological representations on the 3-D SSD.

Let us consider the four orientable meshes shown in Figure 15.22. They represent the same physical object, consisting of 4 faces (denoted by 1, 2, 3 and 4). A VRML viewer will display them identically. Nevertheless, from a geometrical point of view, these meshes are completely different. The mesh in Figure 15.22(a) corresponds to a minimal representation, in the sense that it is composed of a single connected component[4]. In the mesh shown in Figure 15.22(b), each face is a connected component. Meshes in Figure 15.22(c) and Figure 15.22(d) are made of two connected components, defined

[4] A sub-set of mesh faces is *connected* if between each two component faces can be found a path of successive *E-neighbor* faces. A *connected component* is a maximal connected sub-set that can be defined with the faces of the mesh.

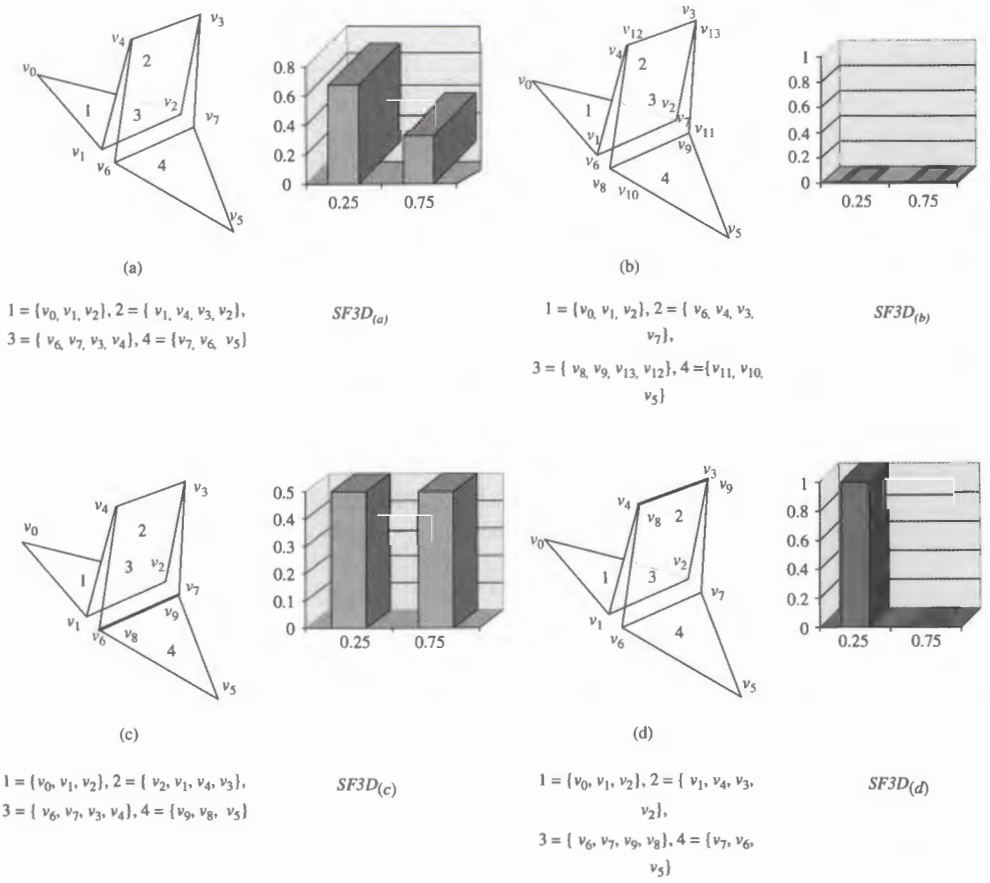


Figure 15.22 Four different topological representations of the same object (thicker line-segments delineate the connected components) and their associated 3-D SSDs

by the faces (1, 2 and 3) and (4) and faces (1, 2) and (3, 4), respectively. Analyzing such surfaces in terms of shape index (or more generally, in terms of local characteristics) will yield completely different results, as shown in Figure 15.22.

In addition, the possible presence of duplicated or degenerated polygons (polygons of zero area) also affects both the global (such as gravity center, axes of inertia, statistical moments) and local characteristics of the considered 3-D surface.

In order to make the 3-D SSD insensitive to the multiple topological mesh representation, it is mandatory to associate with each 3-D object a canonical mesh corresponding to a *minimal and regular topological representation*. Such a canonical mesh has therefore a minimal number of connected components and no degenerated or duplicated polygons. This is achieved by applying a regularizing filtering [14, 15] to the initial 3-D mesh models. Related approaches for repairing 3-D mesh models to form manifold surfaces and solid regions are reported in [16–18].

15.6.4 Example Similarity Measure

The similarity measure is l_1 or l_2 norm computed on vectors formed by the SSD histograms. Such similarity measures may or may not, take into account the nonplanar surfaces or the singular surfaces, depending on the target application.

15.6.5 Experimental Results

The 3-D SSD has been evaluated on a data set consisting of 1290 3-D meshes, where a subset consisting of 228 models has been selected and categorized into 15 distinct categories, on the basis of the perceived shape similarity.

The l_1-based norm was used and the singular and planar histogram bins have not been taken into account. The 3-D SSD has been renormalized such that $\sum_{i=1}^{N_{\text{bins}}} \text{SSD}(i) = 1$.

Figure 15.23 shows some examples of similarity retrieval corresponding to queries from the 'Cars' (Figure 15.23a), '4-Limbs' (Figure 15.23b) and 'Trees' (Figure 15.23c) categories. The first six retrieved results are presented according to a decreasing similarity order, from left to right and from top to bottom. In the upper left cell the query is presented. Figure 15.23(b) shows that good results are obtained for articulated objects in the '4-Limbs' category, in which elements in various postures are retrieved.

For objective evaluation of the proposed 3-D SSD in terms of retrieval performances, the BEP score is computed. After computing the BEP scores for each categorized item, an average BEP is computed for each category. The global mean score of 85% demonstrates the good discriminatory properties of the 3-D SSD.

In order to decrease as much as possible the complexity of the 3-D SSD, the effects of uniformly quantizing the 3-D SSD and of decreasing the number of histogram bins of the representation, with respect to the similarity retrieval-related performances were studied. Experimental results show that using 12 bits for the quantization values yields performance rates approaching the case when floating-point representation is used. More compact representations, down to 8 to 7 bits are also possible with 3% degradation of the mean retrieval rate. Concerning the number of bins of the representations, good results are obtain in a range varying from 100 down to 10 histogram bins, with a degradation of the mean retrieval rate of only 5% [20]. Thus, a compact representation of the 3-D SSD can be used while ensuring optimal retrieval rates.

15.7 EXAMPLE APPLICATIONS OF THE MPEG-7 SHAPE DESCRIPTOR

We will now consider two example applications of MPEG-7 SDs: a cartoon search engine and a trademark search system.

15.7.1 Cartoon Search Engine

Figure 15.24 shows an example of a Web-based cartoon clip search application, which uses MPEG-7 visual descriptors. Short cartoon stories are stored in MPEG-4 format on a

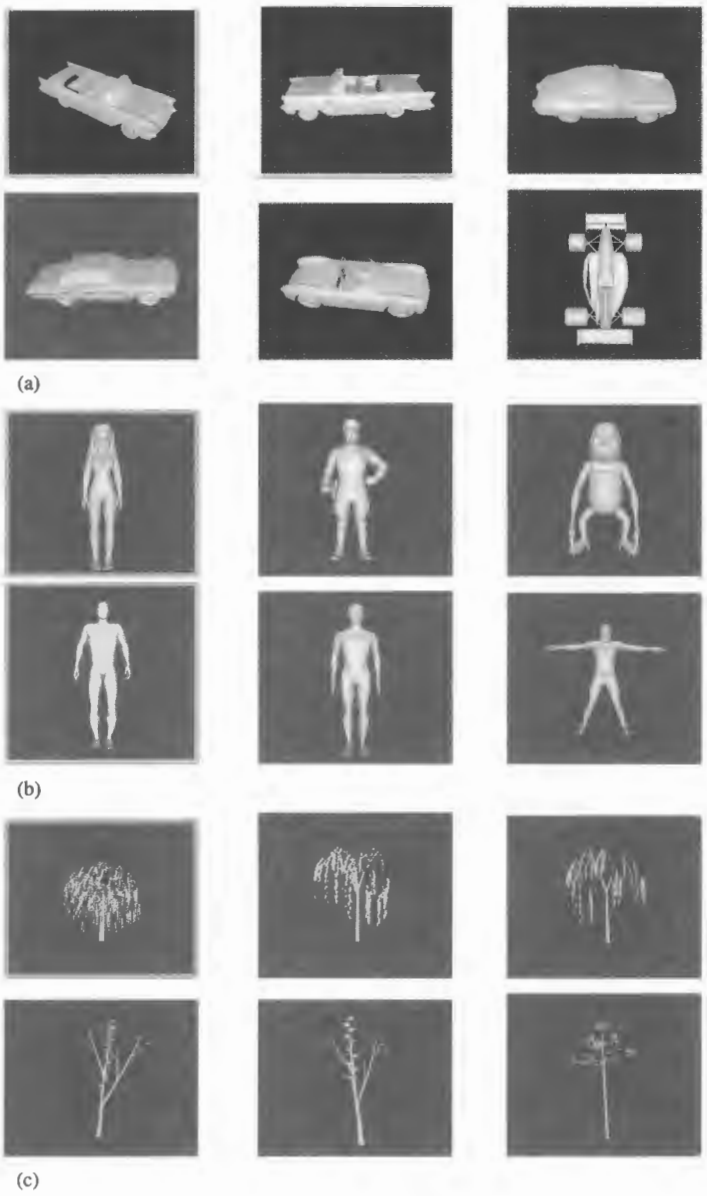


Figure 15.23 Similarity retrieval results for queries within the (a) 'cars'; (b) '4-Limbs'; and (c) 'Trees' categories

server. The characters present in each key frame are segmented and their contour shape and dominant color descriptors extracted and stored as the MPEG-7 descriptors. The user selects a required cartoon character from the group of images in the upper part of the display. Alternatively, the visual properties of the searched character can be drawn or selected with a mouse. A search is performed on the descriptions stored on the servers and key frames from the video clips containing similar characters are displayed to the user

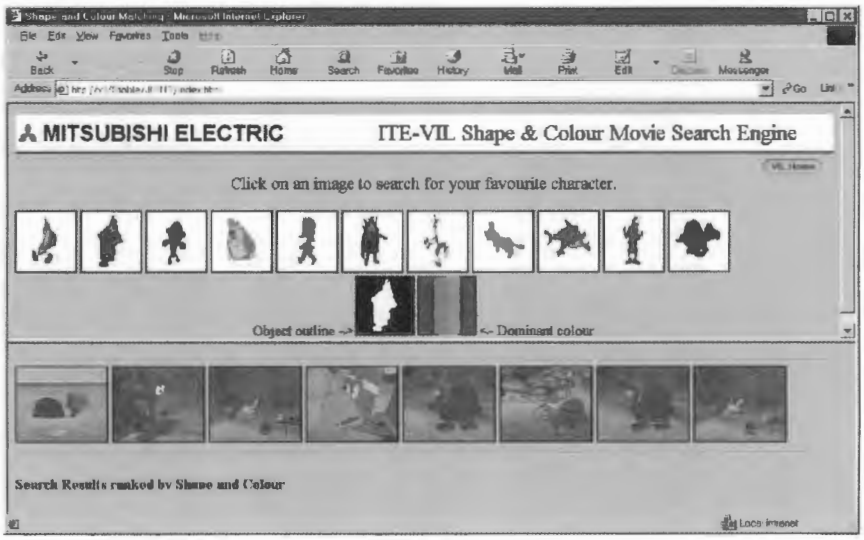


Figure 15.24 Graphical User Interface (GUI) of the shape and color cartoon search engine

(lower row of images). The user can then select the required clip, which is transmitted over the network and played back. Combination of the dominant color and contour-SD gives robust performance, even in the case of partial occlusion, as can be seen in two of the retrieved key frames in the Figure 15.24.

15.7.2 An Application of Region-Based Shape Descriptor for Retrieving Logos

A registered trademark is legally protected from misuse or imitation. The total number of registered trademarks is over a million and the task of designing and registering new trademarks becomes very difficult, as all existing trademarks have to be searched and compared with the new design to avoid unintentional infringement of the copyright. Currently, an operator performs such searches manually – a very tedious and time-consuming task. Automated, similarity-based trademark retrieval system is very useful in this context and an experimental system has been designed using the region-SD [21].

In this experimental system, a query is made to the database either by providing a pictorial example or by a modification to an existing trademark. The system retrieves a set of similar trademarks from the database containing more than 3000 trademarks.

Figure 15.25 shows example retrieval results. Trademarks in the left most column are the query trademarks and the rest on each row are the corresponding top nine retrievals, based on the similarity distance. Trademarks are retrieved regardless of their orientation, thanks to rotational invariance of the region-SD (see TM3 query).

15.8 CONCLUSION

A coherent set of SDs has been developed as an international collaborative effort and now forms a key part of the MPEG-7 visual specification. Extensive tests show that



Figure 15.25 Trademarks in the left-most column are the query trademarks and the rest on each row are the corresponding top nine candidates

the descriptors offer state-of-the-art retrieval performance and have excellent generalization properties, similar to that of the human visual system. High compactness and expression efficiency combined with fast extraction and matching should satisfy even the most demanding real-time application. We expect to see a wide range of novel devices and applications, based on the MPEG-7 shape description technology in the near future. Some very early examples were presented here, but it appears that there is much more to come.

ACKNOWLEDGMENTS

The authors of this chapter would like to thank all participants in the MPEG 'shape experimentation group', especially Y.-S. Kim, K. Muller, J. Ohm, J. Cooper, T. Zaharia, for their technical contributions toward development of the MPEG shape descriptors.

REFERENCES

[1] S. Ullman, *High Level Vision*, The MIT Press, Cambridge, USA, 1997.

[2] ISO/IEC 15938-3: 2001, "Multimedia content description interface – Part 3 Visual," Version 1.

[3] T. Zaharia, F. Preteux and M. Preda, "3D Shape Spectrum Descriptor", TR#15-03, October 1999.

[4] W.-Y. Kim and Y.-S. Kim, "A New Region-Based Shape Descriptor," TR#15-01, December 1999.

[5] F. Mokhtarian and A. K. Mackworth, "A theory of multiscale, curvature-based shape representation for planar curves," *IEEE Transaction On Pattern Analysis and Machine Intelligence*, **14**(8), 789–805 (1992).

[6] J. G. Leu, "Shape normalization through compacting," *Pattern Recognition Letters*, **10**, 243–250 (1989).

[7] F. Mokhtarian and M. Bober, *"The Curvature Scale Space Representation: Theory, Applications, and MPEG-7 Standardization"*, Kluwer Academic Publishers, to appear in 2002.

[8] K. Müller, J.-R. Ohm, J. Cooper and M. Bober, "Results of 2D/3D Shape Core Experiment MS-4," July 2000.

[9] J. Koenderink, *"Solid Shape,"* The MIT Press, Cambridge, Mass., 1990.

[10] M. Do Carmo, *"Differential Geometry of Curves and Surfaces,"* Prentice-Hall, Englewood Cliffs, N.J., 1976.

[11] C. Nastar, "The Image Shape Spectrum for Image Retrieval", INRIA Technical Report RR-3206, July 1997.

[12] C. Dorai and A. K. Jain, "Shape spectrum-based view grouping and matching of 3D Free-form objects," *IEEE Transaction on PAMI*, **19**(10), 1139–1145 (1997).

[13] C. T. Loop, "Smooth Subdivision Surfaces Based on Triangles," Master's Thesis, Department of Mathematics, University of Utah, Salt Lake City, Utah, USA, 1987.

[14] M. Spivak, *"A Comprehensive Introduction to Differential Geometry,"* 2nd ed., Perish, Houston, 1979.

[15] A. Gueziec, G. Taubin, F. Lazarus and W. Horn, "Converting sets of polygons to manifold surfaces by cutting and stitching," *IEEE Visualization '98*, **7**(2), 136–151 (1998).

[16] T. M. Murali and T. A. Funkhouser, "Consistent solid and boundary representations", *Computer Graphics (1997 SIGGRAPH Symposium on Interactive 3D Graphics)*, Providence, RI, ACM, 1997, pp. 155–162.

[17] G. Butlin and C. Stops, "CAD data repair", *Proceedings of the 5th International Meshing Roundtable*, Sandia National Laboratories, 1996, pp. 7–12.

[18] G. Barequet and S. Kumar, "Repairing CAD models," *Proceedings of IEEE Visualization '97*, Phoenix, AZ, USA, October 1997, pp. 363–370.

[19] C.-C. Ten and R. T. Chin, "On image analysis by the methods of moments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10**(4), 496–513 (1988).

[20] T. Zaharia and F. Preteux, "The Influence of the Quantization Step on the 3D Shape Spectrum Descriptor Performances", TR#15-04, July 2000.

[21] Y.-S. Kim and W.-Y. Kim, "Content-based trademark retrieval system using visually salient features", *IEEE Conference on Computer Vision and Pattern Recognition*, San Juan, Puerto Rico, 1997, pp. 307–312.

DVD This document is available on the DVD

16

Motion Descriptors

Sylvie Jeannin[1], Ajay Divakaran[2] and Benoit Mory[3]

[1]*Philips Research, USA,* [2]*Mitsubishi Electric Research Laboratories, USA,*
[3]*Philips Recherche, France*

16.1 INTRODUCTION

The motion features of a video sequence provide the easiest access to its temporal dimension and are hence of key significance in video indexing. Camera motion – estimation techniques, trajectory-matching techniques and aggregated motion vector histogram–based techniques form the core of past work in video indexing using motion characteristics [1–14].

A huge volume of information is required to express the motion field of a typical video sequence. The main aim of motion-based indexing in general and of MPEG-7 in particular is to capture essential motion characteristics into concise and effective descriptors. To be included in MPEG-7, the descriptors also had to comply with the requirements defined by the standard. They had to be generic, not specific to a narrow application or context and had to demonstrate efficiency in enabling wide enough identified applications. They had to be compact in size while providing an expressive representation of the motion feature they characterize. Their extraction from video content had to be simple, as well as their usage for enabling their target applications. Finally, in order to enforce interoperability, the descriptors that passed these tests were further selected so that for one type of usage context and media feature, only one MPEG-7 motion descriptor would be adequate.

As a result of this process, MPEG-7 standardizes four motion descriptors: *Motion Activity*, *Camera Motion*, *Motion Trajectory* and *Parametric Motion*. Some of their applications are:

1. Content-based querying and retrieval from video databases: Motion descriptors enable content-based querying of video content [15–17]. When used in combination with color or texture features, motion indexing significantly improves the performance of similarity-based video retrieval systems (see [3] and the DVD demonstration).

2. Video browsing: Motion descriptors can help find the most active parts of a soccer game, or in basketball actions where the ball is moving toward team A's territory or away from it or where the ball has been kicked high in the air. Thus, for content that is rich in motion, motion indexing enables intelligent browsing.

See DVD for a demonstration of a video retrieval system.

3. Surveillance: Motion descriptors enable event detection in stored or live video. They can be used to trigger alarms when suspicious behavior is detected (fast speed, intrusion in protected area, etc).

4. Video Summarization: In [18] the authors show that motion descriptors can indicate the 'summarizability' of a video sequence. In general, motion descriptors can be used for intelligent summary creation.

5. Video repurposing: The motion descriptors can be used to control the presentation format of content, for example, by dropping more frames when the motion activity intensity is low and fewer when it is high or by slowing down the replay frame rate when the camera motion is a fast pan and so on.

6. Video hyperlinking: The motion descriptors can be used to provide hyperlinks from a region in a video sequence to additional or complementary multimedia content associated with the region [19].

This chapter will present in detail the four MPEG-7 motion descriptors: *Motion Activity*, *Camera Motion*, *Motion Trajectory* and *Parametric Motion*. It is organized as follows. Section 16.2 first summarizes some basics on motion, which are important for better understanding the rest of the chapter. Section 16.3 gives an overview of the motion descriptors standardized by MPEG-7. It highlights their goals and differences and their positions with respect to each other. Then, in Sections 16.4 to 16.7, each of the four descriptors is presented in detail. We describe their syntax and precise semantics, the algorithms needed to use them and their main associated applications and conditions of usage. Finally, a short conclusion summarizes the chapter.

16.2 MOTION BASICS

MPEG-7 deals with visual features extraction and representation. In this section we provide background on motion analysis and representation, which is relevant to the MPEG-7 motion descriptors set.

16.2.1 Motion Analysis

The main purpose of motion analysis is to help develop computer vision systems that can navigate, track, and recognize objects, and estimate their speed and direction (see References 20 and 21 for detailed reviews). While the motion of the objects in the real world is three-dimensional (3-D), capturing a video of their motion results in 2-D projections of the 3-D motion onto the frames of the video sequence. Video motion analysis therefore typically consists of two steps. First, *extraction of 2-D motion parameters* from the frames of the video sequence and second, *extraction of 3-D motion parameters* such as speed from the 2-D motion obtained from step 1. In the following text we will elaborate on step 1. Readers are referred to Reference 22 for further detail on step 2.

There are two broad categories of approaches to step 1, namely, feature correspondence and optical flow. In feature correspondence, a few key features of the frames are first extracted. For instance, in a video of a running horse, we could use the head and tail of the horse as features. Then a correspondence is established between the features of all the frames in the sequence. Even though the aforementioned two steps seem logical and

intuitive, it turns out that they are both nontrivial and difficult to realize. Hence practical schemes rarely use feature correspondence.

The optical flow approach consists of computing a velocity vector for each of the pixels in the frame, forming a dense motion field or an 'optical flow field.' This field lends itself to convenient computation of the 3-D motion. Since dense motion fields require intense computation, practical systems rely on sparser motion fields in which each velocity vector is associated with a moving region or block that consists of several pixels. Therefore, we can further divide the optical flow–based motion analysis into two categories, pixel-based and block-based.

Pixel-based motion estimation

As mentioned above, pixel-based motion estimation involves computation of a motion vector for every pixel. Two well-known approaches are pel-recursion and optical flow. Pel-recursion consists of minimizing the dissimilarity between a small region around the pixel and its counterpart in the previous frame. Pel-recursion thus falls into the category of region-matching techniques.

Optical flow typically combines two kinds of information to provide constraints. The first is conservation information, that is, information that stays the same from frame to frame. An example of such information is the intensity of a pixel under unchanging lighting conditions. The second is neighborhood information, such as uniformity of a connected moving region (smoothness of the motion field) and so on. Incidentally, the well-known Horn-Schunck [23] algorithm makes use of the examples described here. The constraints are combined in a single objective function, which is then minimized.

Optical flow is known to provide highly accurate motion estimation. However, it fails when confronted with variable lighting conditions and occlusion, that is, appearance of a new object or disappearance of an object. It is also highly vulnerable to noise. Optical flow is thus a powerful tool with known limitations. Its applications still remain a topic for active research [24].

Block matching

Block matching is a simple and effective technique for block-based motion analysis and is at the core of all international video standards such as MPEG-1/2/4 [25, 26], H.263 and so forth. In this approach, the frame is first divided uniformly into nonoverlapping rectangular blocks. Because the motion of each block is assumed to be a translation with respect to the previous frame, the motion vector for each block B is computed by finding the best-matching block B' of the same dimensions in the previous frame and computing its displacement from the identically located counterpart of B in the previous frame (see Figure 16.1).

While modeling all motion as translational seems simplistic, it turns out that this approximation works well when the blocks are small enough. In other words, such block-based motion estimation gives satisfactory motion compensation, is computationally simple and yields a sparse motion field that can be represented compactly. It is for these reasons that all practical video-coding schemes use block matching since they are concerned more with efficient encoding of the content than with accurate tracking of object motion.

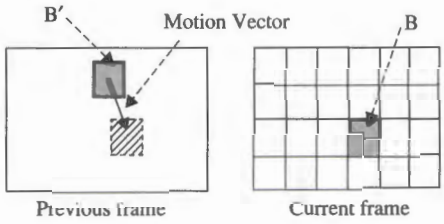


Figure 16.1 Block matching process

16.2.2 Motion Representation

Motion models

As explained above, motion can be represented as a dense, pixel-based motion field. For coding purposes, a sparser, block-based field is usually used, where the underlying model assumes that each block represents a planar surface parallel to the image plane undergoing a translation parallel to the image plane.

For applications other than video coding, this approximation, however, turns out to be too crude. Below are two ways to enhance the simple model. The first is to discard uniformly sized blocks in favor of arbitrarily shaped but homogeneous moving regions. The second is to use higher order models, which allow representing rotation and scaling, affine transformation, planar perspective and parabolic motion of a surface that can be nonparallel to the image plane and nonplanar [22]. The foregoing enhancement constitutes parametric motion estimation in which the motion of regions is estimated by computing the parameters of the model that best matches the motion of the region.

Global motion versus object motion

Motion models should be adapted to the nature of the motion they describe. It is common to consider animated visual data as more or less complex scenes, with a background (moving or not) and eventually some foreground objects animated with their own movements. Background and object movements are different in nature.

Background motion corresponds to the motion of the viewpoint, which is often the camera. It can usually be satisfactorily approximated using parametric motion models. One classical application of this is the generation of sequence panoramas (also called *sprites* or *mosaics*), in which each image is warped on the panoramic image using the transform defined by the parametric motion model (see Figure 16.2). Panoramas can be used for visual browsing or for efficient coding where the entire panorama is coded once and the background of each image is then reconstructed from it using only the motion parameters that define the transform for the current image.

Object motions are much more diverse than background motions and can include complex deformations.

As this section suggests, there is in fact no universal representation for motion. Each application should define its own representation, on the basis of the know-how that

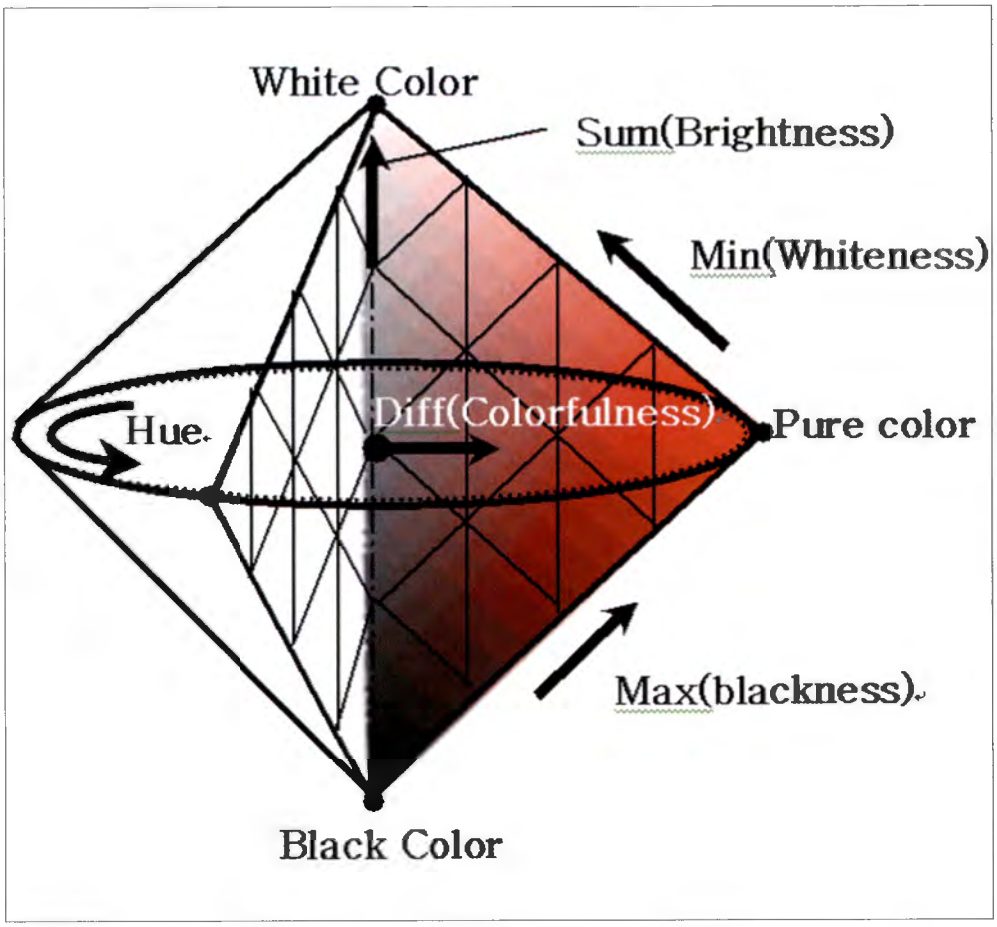


Plate 1 Double cone representation of the HMMD color space

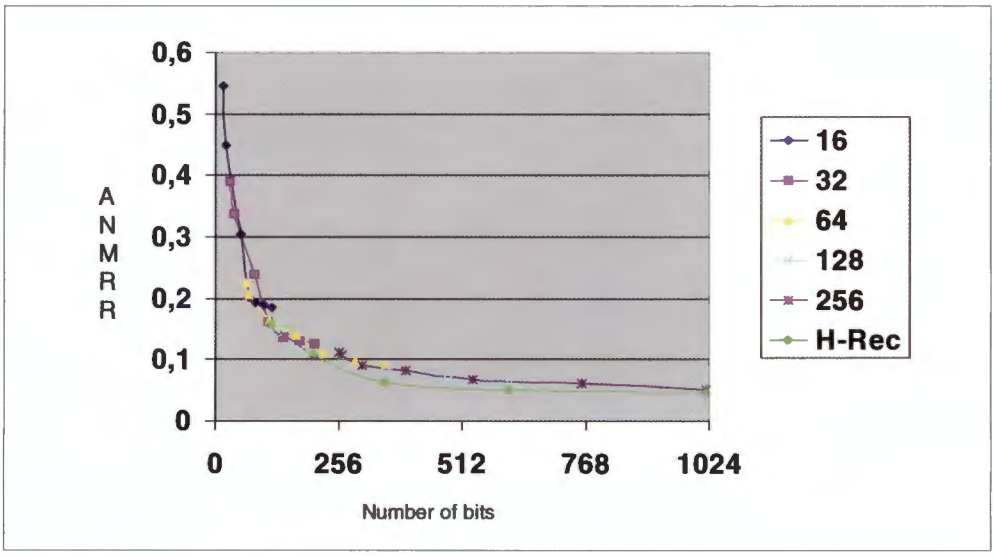


Plate 2 Retrieval results with different number of Haar coeffiecients (16–256) quantized at different number of bits. H-Rec signifies retrieval results after reconstruction of histogram from Haar coefficients at full-bit resolution, which constitutes a lower limit of the efficiency curve

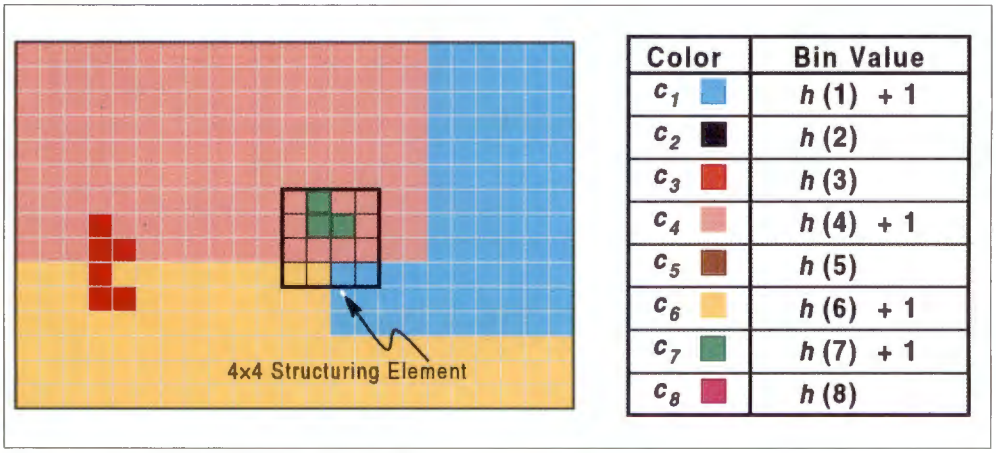


Plate 3 Accumulation of CS Histogram

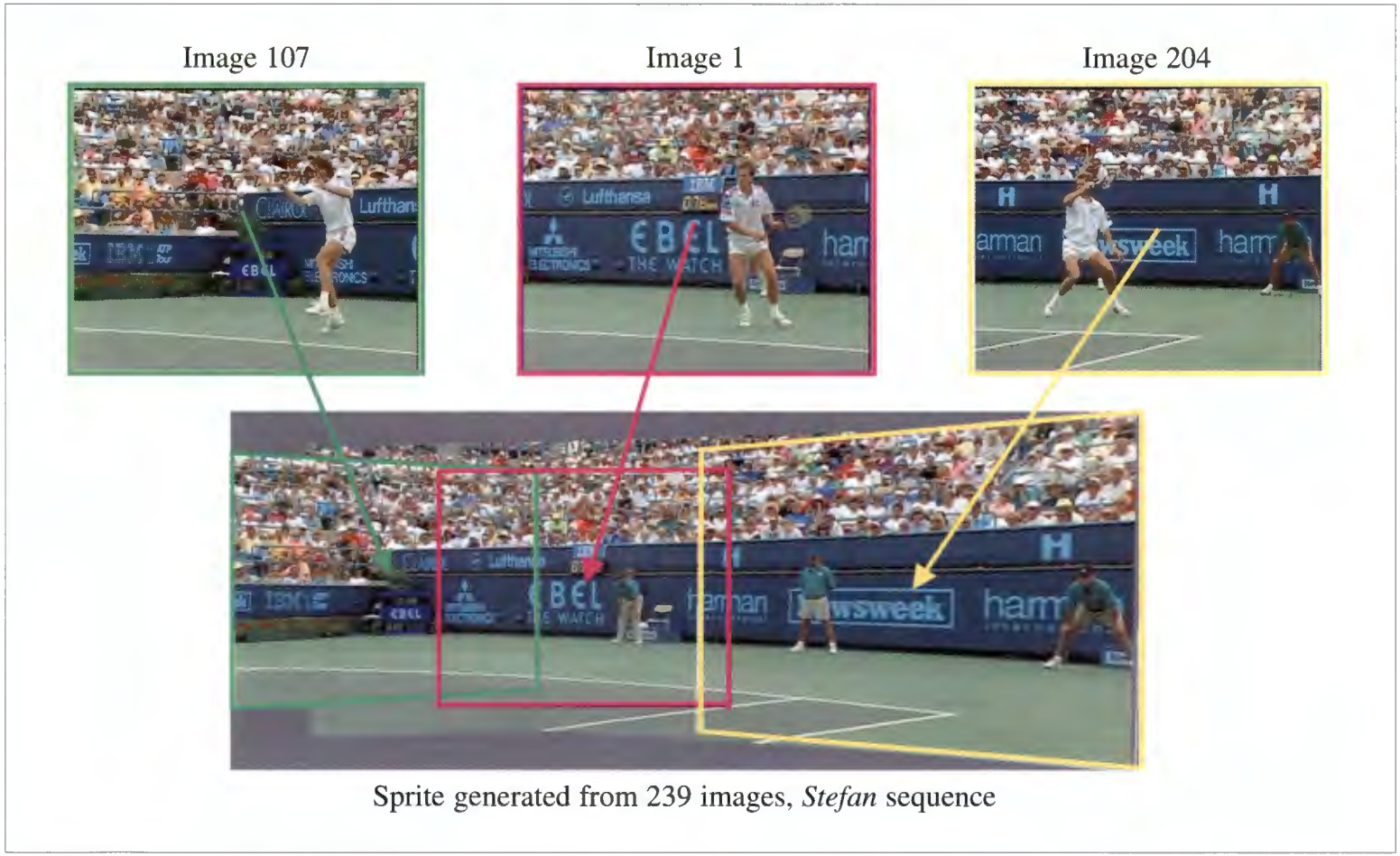


Plate 4 Panorama (or sprite) generation using planar perspective motion model

Ex.1010 / Page 292 of 399
APPLE INC.

16.3.1 Motion Characterization for Video Segments

The overall activity or pace of motion in a segment, which often denotes its level of action, is captured by *Motion Activity*. It is a fairly high-level descriptor, simple to extract from compressed content, which is especially well suited for content browsing or repurposing, as segments with high motion activity are more likely to contain more information. *Motion Activity* contains information on the global intensity of motion activity on a scale from 1 to 5. It can optionally refine this with extra information on (1) its main direction, and/or (2) its spatial localization in the images, and/or (3) its temporal distribution throughout the video segment and/or (4) the type of active regions (one big active region versus many small ones).

Another important concept for describing the motion in a video segment is the movement of the camera. This is described by *Camera Motion*. Camera motion often mimics viewers' focus of attention and can thus be used in certain contexts to discriminate events (e.g. actions in a basket game) or to refine tables of content [17]. *Camera Motion* expresses which type(s) of camera motion is/are present in the segment among all possible camera motion types. It also gives their amplitude and either their precise time localization in the segment or their relative importance in terms of duration (e.g. zoom in 30% of the time).

At a lower level, global motion can also be approximated using successive 2-D parametric transforms. A descriptor called *Warping Parameters* captures these. Typically, these transforms can be used to relate each frame to the segment's panorama or mosaic, which has been largely studied in the literature [27, 28], and recently standardized by MPEG-4 [26, 29] using the terminology 'sprite'. The transmission of the transforms enables functionalities related to mosaics to be provided at the user end, while avoiding their complex computation. *Warping Parameters* contains the coefficients of the parametric motion models (translational, pan-and-zoom, affine, perspective or parabolic) that are used to define the panorama-to-frame relation [26].

16.3.2 Motion Characterization for Moving Regions

The overall displacement of a region, namely, the evolution of its center positioning in time and space (deformations not included) is captured by *Motion Trajectory*. The trajectory of an object is an important component of its behavior, which can help discriminate between objects (human versus car) or types of actions (tennis rallies at the net or not). It gives access to object position, speed and acceleration, typically useful to trigger alarms in surveillance context. *Motion Trajectory* contains the successive 2D or 3D positions of one representative point of the region, such as its center of mass, in time and space, on each spatial dimension independently. Optionally, acceleration information allows interpolating the trajectory between these positions. Different units and spatial coordinate systems, regardless of whether they are dependent on the viewpoint or not, can be used to express positions.

More detailed information about the region displacement is captured by *Parametric Motion*. They help us retrieve objects of similar motions undergoing rotations or deformations, which are not captured by *Motion Trajectory*. *Parametric Motion* uses the same motion model and syntax as the *Warping Parameters* described in Section 16.3.1.

In the following, we present each of these descriptors in more detail. We describe their syntax and precise semantics, elaborate on the algorithms needed to use them, typically for extraction and browsing or retrieval, and emphasize some of their applications and typical conditions of usage.

16.4 MOTION ACTIVITY

A human watching a video or animation sequence perceives it as a slow sequence, fast-paced sequence, action sequence and so forth. The activity descriptor [30] captures this intuitive notion of 'intensity of action' or 'pace of action' in a video segment. Examples of high 'activity' include scenes such as 'goal scoring in a soccer match', 'a high-speed car chase', while examples of low action shots are 'news reader shot', 'an interview scene', and so on. Video content in general spans the gamut from high to low activity. This descriptor enables us to accurately express the activity of a given video sequence or shot and comprehensively covers the aforementioned gamut. It can be used in diverse applications such as content repurposing, surveillance, fast browsing, video abstracting, video editing, content-based querying, and so forth.

16.4.1 Description

To more efficiently enable the applications mentioned above, we need additional attributes of motion activity. Thus, motion activity includes the following attributes.

Intensity of activity: This is expressed by an integer lying in the range [1–5]. A high value of intensity indicates high activity while a low value of intensity indicates low activity.

Direction of activity: While a video shot may have several objects with differing activity, we can often identify a dominant direction. The Direction parameter expresses the dominant direction of the activity if any. It is expressed by a three-bit integer that has a value corresponding to any of eight equally spaced directions.

Spatial distribution of activity: The spatial distribution of activity [15] indicates whether the activity is spread across many regions or restricted to one large region. It is an indication of the number and size of 'active' regions in a frame. For example, a talking head sequence would have one large active region, while an aerial shot of a busy street would have many small active regions. The spatial distribution parameter is expressed by three integers using a total of 16 bits.

Temporal distribution of activity: The temporal distribution of activity expresses the variation of activity over the duration of the video segment or shot. In other words, it describes whether the activity is sustained throughout the duration of the sequence or whether it is only confined to a part of the duration. The temporal distribution parameter is expressed by five 6-bit integers.

In the following section we detail the extraction process for the intensity attribute of the descriptor. Note that it is the only essential attribute of the descriptor and all the other three are optional. Thus, the descriptor can range from 3 bits to 52 bits in size.

16.4.2 Extraction of Intensity of Motion Activity

Since the intensity of motion activity is a subjective measure of the pace or action in a video segment, we first need to construct a ground truth. Then, we assess the performance of automatically computed measures of intensity of motion activity by finding their fidelity to the ground truth. See References 31 and 32 for detailed investigation of this topic.

Construction of ground truth

Our ground-truth data set consists of 637 video segments chosen from the MPEG-7 test set. The segments consist of shots that are typically a few seconds long and cover a wide variety of genres such as news, sports, drama, surveillance, talk shows and so forth. We attempt to maximize the diversity of content while maintaining a manageable data set size. We use human subjects to rate the intensity of motion activity of each of the segments on a five-notch scale viz. (1) very low intensity, (2) low intensity, (3) medium intensity, (4) high intensity and (5) very high intensity. We then average the ratings across the subjects to arrive at the ground truth for each of the video segments.

Automatic measurement of motion activity intensity

With compressed video, motion vectors provide the easiest approach to the gross motion characteristics of the video segment. Since motion-vector magnitude is an indication of the magnitude of the motion itself, it is natural to use statistical properties of the motion-vector magnitude of macroblocks to measure intensity of motion activity. We find that both the standard deviation and the average of the motion-vector magnitude reasonably match the ground truth after proper quantization. However, we also find that the standard deviation of the motion-vector magnitude provides a slightly better approximation of the ground truth, and hence we use quantized standard deviation of the motion-vector magnitude to compute the intensity of motion activity. See Table 16.1 for the thresholds used for quantization of the standard deviation.

Note that these thresholds can be normalized linearly with respect to frame size [30]. The threshold values have been obtained by optimizing the fidelity to the ground truth. However, they work well for all kinds of content and encoding conditions because of the diversity of the ground-truth data set.

Table 16.1 Quantization thresholds for MPEG-1 Video

Activity value	Range of σ (Std. Dev. of motion vector magnitude)
1	$0 \leq \sigma < 3.9$
2	$3.9 \leq \sigma < 10.7$
3	$10.7 \leq \sigma < 17.1$
4	$17.1 \leq \sigma < 32$
5	$32 \leq \sigma$

16.4.3 Typical Usage

Video browsing

The Motion Activity descriptor enables selection of the video segments of a program based on intensity of motion activity. Figure 16.4 illustrates the retrieval of the 10 highest motion activity intensity segments of a news program from Spanish TV, item V3 of the MPEG-7 test set.

In this case, and with other news programs, such retrieval allows us to immediately locate sports segments. The descriptor can similarly help locate other desired parts of the program such as the 'head and shoulders' segments. Often, such intensity-based retrieval gives results that need to be further refined. In Reference 15 we show how to use the spatial-distribution attribute to further narrow down the search results to segments with similar actions such as 'soccer field', and so on.

Content-based querying of video databases

The *Motion Activity* based querying illustrated above could also be a simple first stage of content filtering for content-based querying applications [33]. We can use *Motion Activity* to separate the high- and low-motion parts of the video sequence and then use color descriptors to refine the search. Since intensity of motion activity is a scalar, it is exceedingly simple to match. Hence, it lends itself well to use as a first-stage content filter. For example, while looking for the news anchor, it would be useful to narrow down the search by filtering out all the shots of higher or lower action from the search space. On the news footage from Spanish television mentioned earlier, we are able to narrow down the search space from 1200 to 100 using the motion activity intensity before using a color descriptor over the reduced search space. We are thus able to speed up the content-based matching by a factor of four or more, compared with directly using the color descriptor on the entire search space.

16.4.4 Further Possibilities and Discussion

The aforementioned applications such as video summarization, surveillance, video editing, content repurposing and so on can all capitalize on the gross motion categorization



Figure 16.4 (see also Plate 5) Video browsing – Extraction of the 10 most 'active' video segments in a news program

obtained with the motion activity descriptor. In short, the motion activity descriptor is easy to extract and compact, hence easy to match, and enables a useful gross motion-based clustering of video content. Furthermore, it is robust to changes in the encoder parameters and other sources of noise.

16.5 CAMERA MOTION

The motion of the camera is of high importance within the MPEG-7 indexing framework since it often relates to the content semantics. Indeed, two video shots of the same real-world scene with different camera movements convey different subjective representations of the reality. More specifically, assumptions can be made on the correlation between high-level characteristics of the content and the motion of the camera. The knowledge of the temporal sequence of camera operations can help discriminate between certain domains or genres such as news, home videos, soccer games, action movies. For movies, they might reflect artistic choices of the director (e.g. shooting from the cameraman's shoulder resulting in wobbly footage or using only fixed shots). In general, rotations (see Figure 16.5b) are used either to provide an overview of a large scene (e.g. a landscape or a skyscraper) or to follow an object in a short period of time. Translations are mostly used to follow moving objects (e.g. travelling) for a longer period of time, while zoom operations often mimic human focus of attention on a particular object and are likely to precede a close-up. Rolling movements (see Figure 16.5b) can be representative of high-dive acrobatic sequences or subjective motorcycle viewpoints during a race.

16.5.1 Description

Camera Motion supports all camera operations such as translations, rotations and changes of focal length together with the possible combinations of these. The eight basic camera operations usually defined (see Figure 16.5a and Figure 16.5b) are panning (horizontal rotation), tracking (horizontal transverse movement), tilting (vertical rotation), booming (vertical transverse movement), zooming (change of the focal length), dollying (translation along the optical axis), rolling (rotation around the optical axis) and finally the absence of operation (fixed camera). All these camera motion operations induce different instantaneous global velocity fields in the image plane that can be either modeled and extracted automatically or generated by capture devices.

Long-term representation

Camera operations can be described for each frame. For each pair of frames of a video sequence, the camera motion parameters can be estimated (qualitatively and quantitatively), for instance using techniques adapted from Reference 34 as described in Reference 35. This frame-based information is relevant for professional applications for which a very high level of details could be required. Nevertheless, for most retrieval applications, a long-term description of the camera motion should suffice. Typically, providing information such as the dominant motion type of each shot may be more relevant and usable than the instantaneous camera parameters. The MPEG-7 camera motion descriptor

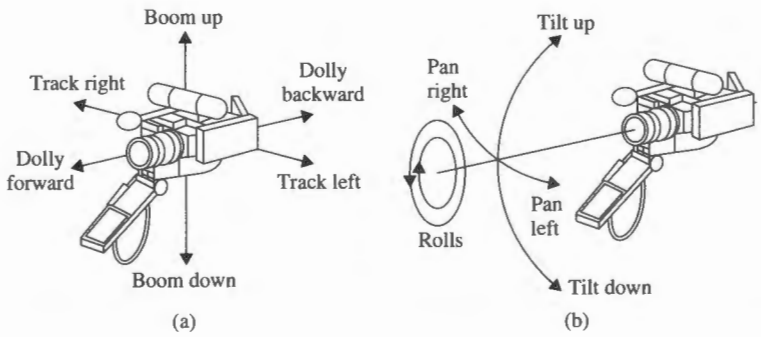


Figure 16.5 Camera operations: (a) track, boom, dolly and (b) pan, tilt, roll

structure is hence designed to handle such a long-term representation (fractional presence along time), together with the ability to provide the finest level of description.

Motion types

From the list of operations presented in Figure 16.5, 15 motion types are defined, taking into account their respective direction. The division of each movement into two directions is needed since opposite camera operations can be successively present and hence jointly described in the same piece of video (e.g. a zoom in is followed by a zoom out). The motion types are: Track (left and right), Boom (up and down), Dolly (forward and backward), Pan (left and right), Tilt (up and down), Roll (clockwise and counterclockwise), Zoom (in and out) and Fixed.

Additional parameters

For each motion type in the video segment being described, the following parameters are defined: fractional presence and amount of motion.

Let *total_duration* be the duration of the time interval chosen for a given description. The *fractional presence* Δ_{motion_type} of a given motion type in this interval is defined by the fraction of the total duration for which the said motion type is present, expressed as a percentage (see Equation 16.1 below).

$$\Delta_{motion\_type} = \frac{duration_{motion\_type}}{total\_duration} \quad (1)$$

The *amount of motion* describes 'how much' of a particular camera motion type (e.g. zoom) there is in the time interval for which the motion type is present. It is defined as the average fraction of the image that was covered (or uncovered) because of the given motion type, expressed as a percentage. The *amount of motion* is a generic and quantitative measure of the global motion induced by any camera operation.

Note that the fractional presence and the amount of motion are designed to be independent of the video-coding format, frame rate (e.g. 25, 30 Hz) and spatial resolution (e.g. Quarter CIF (QCIF), Common Interchange Format (CIF)), which fulfils an important requirement of MPEG-7.

Description syntax (overview)

The core of the description is composed of a set of N camera motion segments ($N = 1$ for a high-level description, $N =$ *total number of frames in the segment* for a frame-based detailed description). Each camera motion segment includes:

- timing information: *start_time* and *duration* parameters are used to localize the segment in time.
- a set of 15 optional *fractional_presence* parameters corresponding to the possible motion types.
- a set of 14 optional *amount_of_motion* parameters (the amount of 'fixed' motion is always equal to 0 and is thus not specified).
- if relevant, the focus of expansion (resp. contraction), which is the imaginary point (expressed in normalized coordinates) toward which all image velocity directions converge (resp. diverge).

16.5.2 Matching

MPEG-7 recommends a matching function dedicated to *Camera Motion*. As the meaning of similarity depends on the exact query purpose, some query criteria that parameterize the similarity function are defined. Two binary parameter vectors (α_I and β_I) specify which fields in the descriptions being compared must be taken into account in the computation of the matching result ($\alpha_i = 1$, $0 \leq i \leq 15$, if the *fractional presence* of motion type i must be taken into account, $\alpha_i = 0$ otherwise.$\beta_i = 1$, $0 \leq i \leq 14$, if the *amount of motion* of motion type i must be taken into account, $\beta_i = 0$ otherwise). In addition, three weights are used to assign a relative importance to the *fractional presence* (w_P), the *amount of motion* (w_A) and the *segment duration* (w_D).

Let $C1$ and $C2$ be two camera motion descriptions associated with two temporal video segments. Let P_i^{C1}, A_i^{C1} (respectively P_i^{C2}, A_i^{C2}) be the fractional presence factors and the amount of motion magnitudes of each motion type i. Let D^{C1} (respectively D^{C2}) be the whole duration of the considered temporal segment. The similarity $M(C1, C2)$ between the two descriptors can be computed as shown in Equation 16.2 below.

$$\begin{cases} M(C1,C2) = \dfrac{w_P M_P(C1,C2,\boldsymbol{\alpha}) + w_A M_A(C1,C2,\boldsymbol{\beta}) + w_D M_D(C1,C2)}{w_P + w_A + w_D} \\ M_P(C1,C2,\boldsymbol{\alpha}) = \dfrac{\sum \alpha_i.[1 - |P_i^{C1} - P_i^{C2}|/(P_i^{C1} + P_i^{C2})]}{\sum \alpha_i} \\ M_A(C1,C2,\boldsymbol{\beta}) = \dfrac{\sum \beta_i.[1 - |A_i^{C1} - A_i^{C2}|/(A_i^{C1} + A_i^{C2})]}{\sum \beta_i} \\ M_D(C1,C2) = 1 - |D^{C1} - D^{C2}|/(D^{C1} + D^{C2}) \end{cases} \quad (2)$$

Note that the so-defined matching function is between 0 (complete mismatch) and 1 (perfect match), allowing similarity-based retrieval as well as the rank-ordering of the query results. Different application-dependent similarity meanings are allowed by the query parameters (e.g. searching for video segments containing strong pans *and* tilts of *any* duration *with or without* zoom).

16.6 MOTION TRAJECTORY

Motion trajectory describes the displacements of objects in time, an object being defined as any spatiotemporal region or set of spatiotemporal regions whose trajectory is relevant in the context in which it is used. It is a fairly high-level descriptor. The successive spatiotemporal positions of the object are represented by the positions of one representative point of the object, such as its center of mass. Such simplicity is relevant, as humans perceive objects motion at a high level [35]. Besides, experiments show that such high-level data are rich enough to enable most relevant applications (video hyperlinking [19], motion-based querying [16], etc).

16.6.1 Description

The trajectory model is a first- or second-order piecewise approximation of the spatial positions of the representative point along time, for each spatial dimension:

- *first-order approximation (default interpolation)*:

$$x(t) = x_i + v_i(t - t_i) \quad \text{with} \quad v_i = \frac{x_{i+1} - x_i}{t_{i+1} - t_i} \tag{3}$$

- *second-order approximation*:

$$x(t) = x_i + v_i(t - t_i) + \frac{1}{2}a_i(t - t_i)^2 \quad \text{with} \quad v_i = \frac{x_{i+1} - x_i}{t_{i+1} - t_i} - \frac{1}{2}a_i(t_{i+1} - t_i) \tag{4}$$

and similarly for the other dimensions y and z. With such a model, v_i and a_i do correspond to the object's speed and acceleration respectively, considered constant on $[t_i, t_{i+1}]$, x_i and x_{i+1} are the positions at times t_i and t_{i+1}.

On the basis of this model, the core of the description is a set of *key points*, representing the successive spatiotemporal positions of the described object, between which trajectory is interpolated. Key points are defined by their coordinates in space (2D or 3D) and time: (x, y, t) or (x, y, z, t). By default, linear interpolation is used (Equation 16.3). When nonlinear interpolation is done between two key points in one spatial dimension, the corresponding interpolating parameter, a_i, is added to specify the second-order function of time that should be used (Equation 16.4).

Figure 16.6 illustrates the trajectory representation. It shows on the left a modeled trajectory, instantiated by the parameters in bold (x_i, t_i, a_0), and on the right the corresponding bitstream structure. In the bitstream, all the times (here t_0, t_1, t_2, t_3) are given first. Then, for each spatial dimension, key point positions with optional interpolating data

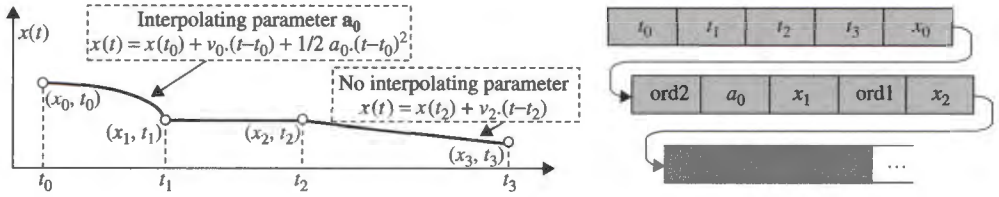


Figure 16.6 Example of trajectory representation (1 dimension) and corresponding bitstream structure

follow (ord1, ord2 specify the type of interpolation used on the current interval: first or second order).

Besides this main core, the syntax also contains:

- A boolean value specifying whether or not the camera follows the object being described. This field is useful because objects followed by the camera always appear still, roughly centered in the image: their trajectory as defined above will not give much more information and is not as useful.
- Information about the units and coordinate system used for expressing positions, together with the precision used to encode the coordinate values.

Description properties

The description size is roughly proportional to the number of key points and depends on the coordinates resolution [36]. Using the flexibility of the syntax, one can choose the number and temporal instances of key points to best adapt the description to the content and/or application. The total number of key points can be chosen so that the global precision and compactness required by the application is met, and variable time intervals between key points can be chosen to match the local trajectory's smoothness. Besides, key points data can be interpreted alone, and interpolation parameters can be considered as an enhancement layer.

16.6.2 Extraction

The input to the extraction is the information defining the object, that is, its spatiotemporal location in the visual data. It can be given or extracted on the fly, as trajectory is instantiated. The format of this information can be an MPEG-4 *alpha channel* [26], a bounding box given by a tracking system, a segmentation mask calculated from the image pixels or the motion vector values [5] or any equivalent data, depending on the application environment.

The trajectory descriptor can be instantiated very easily from these data [30]. Time instants are initialized as successive frame-time instants (one per frame). Then, for each time instant, the positions (x, y) of the trajectory point are instantiated as the coordinates of

the center of mass of the region (mask, bounding box,...), and the interpolating parameters are calculated as the local second-order derivatives of the positions. As a result, a valid MPEG-7 trajectory description is obtained.

Selection of key points and functions

The description can be further adapted to the application, as explained in Section 16.6.1., by subsampling the obtained set of points (i.e. merging adjacent approximation intervals). Different classical methods can be applied for the merging, such as:

- *Algo. 1: sequential*: The system starts with an approximation interval, which is the union of the first two intervals: $[t_0,t_1]U[t_1,t_2]$, made of the first three points. It sequentially adds the following points to expand this interval, until the interpolation error exceeds a given threshold. This interval is kept, and the process repeats from the next three instants.
- *Algo. 2: recursive*: The system starts with an approximation interval containing all the points, and splits the interval into two at the position where the interpolation error is maximum. The splitting process is recursively repeated on each two-new intervals. It stops either when the desired description size is reached or when the error drops below a given threshold.

The algorithm should be chosen on the basis of the constraints of the application. For instance, Algo. 2 can only be used if all spatiotemporal positions are known as it starts. In general, as the complexity of the trajectory extraction resides almost exclusively in the selection process, complexity is also an important criterion in this choice. Note that subsampling at fixed intervals between key points insures a selection at no complexity price.

16.6.3 Usage

Similarity matching

Combining trajectory descriptions with other descriptors to retrieve similar video segments significantly improves retrieval performance. An efficient and simple distance measure $M(T1, T2)$ between two trajectories is a linear weighting of distances between object position M_P, speed M_S, (to be calculated from position and time data) and acceleration M_A, as shown in Equation 16.5 below.

$$M(T1,T2) = \frac{w_P M_P(T1,T2) + w_S M_S(T1,T2) + w_A M_A(T1,T2)}{w_P + w_S + w_A}$$
$$M_P(T1,T2) = \sum_i \frac{(x1_i - x2_i)^2 + (y1_i - y2_i)^2 + (z1_i - z2_i)^2}{\Delta t_i} \quad (5)$$

and with similar definitions for M_S and M_A. i is the index of the time interval and Δt_i is the duration of the ith time interval. The weights (w_P, w_S, w_A) allow to adapt the matching strategy to the application environment or to specific user's interests. Additional weights could also be added on x, y and z components within $M_P M_S$ and M_A to give more importance to some spatial dimension(s) [30].

Higher level queries

Motion Trajectory syntax provides direct and independent access to positions, speeds and accelerations, in each spatial dimension. Many high-level queries can thus be implemented very simply [37]. For instance:

- 'retrieve objects passing near a given area', useful in indoor surveillance contexts, is simply a test on the key points spatial-positions values;
- 'retrieve objects moving faster than a given speed', useful for road monitoring, is a comparison between the speed values, which are straightforward to calculate from the key points positions and time, and the given speed.

Further usages

Another example of trajectory usage is video hyperlinking: linking some predefined objects in videos to other multimedia contents, to enable rich interactive viewing. Using several trajectories to represent the evolution of a few contour points of an object in time has proved to be a very efficient way of enabling this [19].

16.7 PARAMETRIC MOTION

16.7.1 Description

Parametric Motion represents the motion and/or deformation of a region or image by one of the following classical parametric motion models:

- Translational (2 parameters): $v_x(x, y) = a_1$, $v_y(x, y) = a_2$
- Rotation/scaling (4 parameters): $v_x(x, y) = a_1 + a_3x + a_4y$, $v_y(x, y) = a_2 - a_4x + a_3y$
- Affine (6 parameters): $v_x(x, y) = a_1 + a_3x + a_4y$, $v_y(x, y) = a_2 + a_5x + a_6y$
- Perspective (8 parameters): $v_x(x, y) = (a_1 + a_3x + a_4y)/(1 + a_7x + a_8y)$, $v_y(x, y) = (a_2 + a_5x + a_6y)/(1 + a_7x + a_8y)$
- Quadratic (12 parameters): $v_x(x, y) = a_1 + a_3x + a_4y + a_7xy + a_9x^2 + a_{10}y^2$, $v_y(x, y) = a_2 + a_5x + a_6y + a_8xy + a_{11}x^2 + a_{12}y^2$

where $v_x(x, y)$ and $v_y(x, y)$ represent the x and y displacement components of the pixel at coordinates (x, y).

The core of the description specifies which motion model is used among the ones listed above, the time interval on which it is used, the information about the coordinate system and finally the value of each parameter a_i.

16.7.2 Extraction

The issue of parametric motion model extraction has been widely addressed in the literature [22]. It is classically solved as a regular optimization problem, where the variables



are the parameters of the chosen motion model, and the motion compensation error is the objective function to be minimized. The convergence process is, however, sensitive and should be carefully implemented to achieve robustness. A method has already been proposed by MPEG-4, for extracting Sprites [26]. The MPEG-7 recommended method [30] is based on this and is further optimized to achieve robustness in the context of object motion extraction.

The error measure is first defined as $E = \sum_i [\mathrm{I}'(x_i', y_i') - \mathrm{I}(x_i, y_i)]^2$ where $\mathrm{I}'(x_i', y_i')$ is the motion transformed image to be aligned onto the reference image $\mathrm{I}(x_i, y_i)$, and the sum is done over all common pixels.

In order to minimize the error E, an iterative method is used, which at each iteration calculates an appropriate search direction in the parameter space, and then performs a line search in this direction.

The search direction $\overline{d}$ is calculated using the first and second partial derivatives of the error function as $\overline{d} = -(\varepsilon\mathbf{I} + \mathbf{H})^{-1} \cdot \overline{g}$ where $\mathbf{I}$ is the identity matrix, $\overline{g}$ is the gradient of the error function defined as

$$g_k = \frac{\partial E}{\partial a_k} = \sum_i 2 \cdot (\mathrm{I}'(x_i', y_i') - \mathrm{I}(x_i, y_i)) \cdot \left(\frac{\partial \mathrm{I}'(x_i', y_i')}{\partial x'} \cdot \frac{\partial x'}{\partial a_k} + \frac{\partial \mathrm{I}'(x_i', y_i')}{\partial y'} \cdot \frac{\partial y'}{\partial a_k} \right)$$

and $\mathbf{H}$ is the matrix of second derivatives, namely, the Hessian matrix, defined as

$$\mathrm{H}_{kl} = \frac{\partial^2 E}{\partial a_k \partial a_l} = \sum_i \left[\frac{\partial \mathrm{I}'(x_i', y_i')}{\partial a_k} \cdot \frac{\partial \mathrm{I}'(x_i', y_i')}{\partial a_l} + [\mathrm{I}'(x_i', y_i') - \mathrm{I}(x_i, y_i)] \times \left(\frac{\partial \mathrm{I}'(x_i', y_i')}{\partial x'} \cdot \frac{\partial^2 x'}{\partial a_k \partial a_l} + \frac{\partial \mathrm{I}'(x_i', y_i')}{\partial y'} \cdot \frac{\partial^2 y'}{\partial a_k \partial a_l} \right) \right]$$

The derivatives $\partial x'/\partial a_k$ and $\partial y'/\partial a_k$ are easily calculated from the motion models, and $\partial \mathrm{I}'(x_i', y_i')/\partial x'$ and $\partial \mathrm{I}'(x_i', y_i')/\partial y'$ are calculated using gradient operators on the transformed image.

Epsilon is chosen as small as possible on condition that the expression $(\varepsilon\mathbf{I} + \mathbf{H})$ is positive-definite, which means that $(\varepsilon\mathbf{I} + \mathbf{H})$ can be inverted and $\overline{d}$ is a descent direction. One way to determine epsilon is to try performing a Cholesky factorization of the form $(\varepsilon\mathbf{I} + \mathbf{H}) = \mathbf{P}\mathbf{P}^{\mathbf{T}}$. If the factorization fails, epsilon is increased until factorization succeeds. More details about this algorithm can be found in Reference 30.

In order to guarantee the convergence of the iterative process, we can embed it into a multiresolution framework that contain several levels of image scale. Within this framework, the process starts with finding the correct parameter values on a coarse level. The resulting values are then used as initial parameter values on a finer scale, and the minimization is continued. Besides, at each level of scale, a minimization process updating only the translational parameters is performed before the full minimization using all parameters of the chosen motion model is done, which serves as initialization for the minimization on the full model.

Two other important factors for the robustness of this algorithm are to maintain a high precision at each step of the process (filtered images calculations and interpolations for transform calculation) and to discard outliers when calculating the error gradient [30].

16.7.3 Usage

This motion representation allows retrieving objects with similar motion behaviors, especially in contexts wherein objects are likely to undergo complex motions or deformations that cannot be captured by *Motion Trajectory*. Different processes can be used to evaluate similarities, depending on the context of the application.

Given two sets of parametric object motion parameters $\mathbf{a} = (a_1, a_2, \ldots, a_N)^T$ and $\mathbf{b} = (b_1, b_2, \ldots, b_N)^T$, where N is the number of parameters of the applied motion model, a very simple distance measure d can be directly computed in the parameters space, as shown in Equation 16.6 below:

$$d = \sum_{n=1}^{N} w_n \cdot |a_n - b_n| \tag{6}$$

where w_n are weighting factors allowing the individual adjustment of the importance of each motion parameter.

Another distance measure, which is more complex but allows comparing descriptions that use different motion models, consists in going back to the motion vector space. The dense motion vector field corresponding to each motion model is calculated on the union of the two regions, and differences between the motion vectors of each field are added on each point to get the final distance measure value.

16.8 CONCLUSION

MPEG-7 standardizes four motion descriptors: *Motion Activity*, *Camera Motion*, *Motion Trajectory* and *Parametric Motion*. *Motion Activity* characterizes the pace of action in video segments, and *Camera Motion* the camera operations. *Motion Trajectory* describes the displacements of objects. *Parametric Motion* approximates both objects and global motions and deformations at a more detailed lower level. Altogether, they provide a complete, coherent and useful collection of descriptions that capture the different aspects of motion in videos, with a broad- range of precision. They were designed to be concise in size, easy to extract and use.

When used in combination with other features such as color or texture, they significantly improve the performance of similarity-based video-retrieval systems. They also enable motion-based queries, which are useful in contexts in which motion has a rich meaning such as sport or surveillance. Finally, they also enable rich complementary functionalities, such as content repurposing or intelligent summary creation, video hyperlinking, refined browsing, refinement of the table of content and much more. Altogether, they facilitate a wide variety of applications besides the content-based querying of multimedia databases.

As a complement to this chapter, the reader can refer to and install the demonstration software provided in the accompanying DVD. It aims at showing sample results of an MPEG-7 similarity-based retrieval engine using motion features in a video-segments database. It also shows how motion features, when combined with static visual features such as color, can be used to perform low-level but meaningful queries.

ACKNOWLEDGMENTS

We would like to acknowledge the work of all MPEG members, whose list would be too long to give here, who have contributed to the definition of the motion descriptions presented in this chapter and who took part in related core-experiments.

REFERENCES

[1] A. Akutsu *et al.*, "Video indexing using motion vectors," *Proceedings of Visual Communications and Image Processing*, SPIE Vol. 1818, 1522–1530, (1992).

[2] E. Ardizzone *et al.*, "Video indexing using MPEG motion compensation vectors," Proceedings of IEEE International Conference on Multimedia Computing and Systems, Florence, Italy, 1999.

[3] S. F. Chang, *et al.*, "A Fully Automated Content-Based Video Search Engine Supporting Multi-Objects Spatio-Temporal Queries," *IEEE Transactions on Circuit and Systems for Video Technology*, **8**(5), 602–615 (1998).

[4] J. W. Davis, Recognizing movement using motion histograms, Technical Report No. 487, MIT Media Laboratory Perceptual Computing Section, Cambridge, Mass., April 1998.

[5] N. Dimitrova and F. Golshani, "Motion recovery for video content analysis," *ACM Transaction on Information Systems*, **13**(4), 408–439 (1995).

[6] V. Kobla, D. Doermann, K.-I. Lin and C. Faloutsous, "Compressed domain video indexing techniques using DCT and motion vector information in MPEG video," *Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Databases V*, SPIE Vol. 3022, 200–211, (1997).

[7] W. Niblack, D. B. Ponceleon and D. Petkovic *et al.*, "Updates to the QBIC System," *Proceedings IS&T SPIE, Storage and Retrieval for Image and Video Databases VI*, Vol. 3312, San Jose, 1998, pp. 150–161.

[8] E. Sahouria, Video Indexing Based on Object Motion, M.S. thesis, Department of EECS, University of California, Berkeley, 1997.

[10] Y. P. Tan, S. R. Kulkarni and P. J. Ramadge, "Rapid estimation of camera motion from compressed video with application to video annotation," *IEEE Transaction on Circuits and Systems for Video Technology*, **10**(1), 133–146 (2000).

[11] Y. T. Tse and R. L. Baker, "Camera zoom/pan estimation and compensation fore video compression," *Proceedings of SPIE Conference On Image Processing Algorithms and Techniques II*, Boston, Mass., 1991, pp. 468–479.

[12] *http://www.virage.com*.

[13] *http://www.ctr.columbia.edu/videoq*.

[14] *http://www.qbic.almaden.ibm.com*.

[15] A. Divakaran and H. Sun, "A Descriptor for spatial distribution of motion activity," *Proceedings of SPIE Conference on Storage and Retrieval from Image and Video Databases*, San Jose, Calif., 2000, pp. 24–28.

[16] S. Jeannin and B. Mory, "Video Motion Representation for Improved Content Access," *IEEE Transactions on Consumers Electronics*, **46**(3), 645–655 (2000).

[17] J. Llach and P. Salembier, "Analysis of video sequences: table of contents and index creation," *Proceedings of International Workshop on Very Low Bit-Rate Video Coding (VLBV'99)*, Kyoto, Japan, 1999, pp. 52–56.
[18] A. Divakaran, R. Radhakrishnan and K. Peker, "Video summarization with motion descriptors," *Journal of Electronic Imaging*, **10**(4), 909–916 (2001).
[19] O. Hori and T. Kaneko, "Results of Spatio-Temporal Region Descrition Scheme Core/Validation Experiment," *ISO/IEC JTC1/SC29/WG11/MPEG99/M5414*, Dec. 1999, Maui.
[20] Y. Q. Shi and H. Sun, *"Image and Video Compression for Multimedia Engineering, Fundamentals, Algorithms and Standards,"* CRC press, Boston, Fla., 2000.
[21] G. Tziritas and C. Labit, "Motion analysis for image sequence coding," *Advances in Image Communication*, Vol. 4, Elsevier, Amsterdam, The Netherlands, 1994.
[22] J. L. Dugelay and H. Sanson, "Differential methods for the identification of 2D and 3D motion models in image sequences," *Signal Processing, Image Communication*, **7**, 105–127 (1995).
[23] K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, **17**, 185–203 (1981).
[24] I.-J. Lin, Video Object Plane Extraction and Representation: Theory and Application, Ph.D. thesis, Department of Electrical Engineering, Princeton University, June 2000.
[25] "Information Technology, Coding of moving pictures and associated audio for digital storage media at up to 1,5 Mbits/s (MPEG-1), Part 2: Video," *ISO/IEC* 11172-2: 1993.
[26] ISO/IEC 14496-2, "Information technology, Coding of Audio-visual Objects (MPEG-4), Part 2: Video," 1999.
[27] R. Szeliski, "Image mosaicing for tele-reality applications," *IEEE Workshop on Applications of Computer Vision*, IEEE Computer Society, 1994, pp. 44–53.
[28] M. Irani and P. Anandan *et al.*, "Efficient representation of video sequences and their application," *Signal Processing: Image Communications*, **8**, 327–351 (1996).
[29] *http://www.cselt.it/mpeg/*.
[30] ISO/IEC 15938–8:2002, "*Multimedia Content Description Interface – Part 8: Extraction and use of MPEG-7 Descriptions*," version 1, 2002.
[31] A. Divakaran, H. Sun, H. Kim, C. S. Park, B. S. Manjunath, X. Sun, H. Shin and V. V. Vinod *et al.*, "Report on the MPEG-7 CE on the Motion Activity Feature," ISO/IEC MPEG99/m5030, Melbourne, Australia, Oct. 2000.
[32] K. Peker and A. Divakaran, "Automatic measurement of intensity of motion activity of video segments," *Proceedings of SPIE Conference on Storage and Retrieval from Multimedia Databases*, San Jose, Calif., 2001.
[33] A. Divakaran, A. Vetro, H. Nishikawa and K. Asai, "Video browsing system based on compressed domain feature extraction," *IEEE Transactions on Consumer Electronics*, 2000, pp. 637–644.
[34] M. V. Srinivasan, S. Venkatesh and R. Hosie, "Qualitative estimation of camera motion parameters from video sequences," *Pattern Recognition*, **30**(4), 593–606 (1997).
[35] D. Marr, *Vision*, W.H. Freeman, San Francisco, 1982.
[36] K. W. Lee, W. S. You and J. Kim, "Quantitative analysis for the motion trajectory descriptor in MPEG-7," *ISO/IEC JTC1/SC29/WG11/MPEG99/M5400*, Dec. 1999, Maui.
[37] S. Jeannin, R. Jasinschi and B. Mory *et al.*, "Motion descriptors for content-based video representation," *Signal Processing: Image Communication Journal*, **16**(1,2), 59–85 (2000).
[38] W. Wolf, "Key frame selection by motion analysis," *Proceedings of ICASSP 96*, **II**, 1228–1231 (1996).

Section V
AUDIO

17

Fundamentals of Audio Descriptions

Adam T. Lindsay[1], Ian Burnett[2], Schuyler Quackenbush[3] and Melanie Jackson[4]

[1]*University of Lancaster, UK,* [2]*University of Wollongong, NSW, Australia,*
[3]*AT&T Bell Labs, USA,* [4]*University of Wollongong, NSW, Australia*

This chapter gives an overview of the emerging MPEG-7 standard for audio metadata: ISO/IEC 15938-4 Information Technology – Multimedia Content Description Interface – Part 4 Audio [1]. Specifically, this overview details the structure of MPEG-7 Audio, the applications it may support and its constituent descriptors and description schemes. At the beginning of 2002, the standard was at the Final Draft International Standard (FDIS) stage, and is expected to be an International Standard (IS) now. However, this work in progress will be referred to as the MPEG-7 Audio standard.

17.1 THE STRUCTURE OF THE STANDARD

MPEG-7 Audio includes a foundation layer of low-level tools that apply to any sound in a general way. These tools establish a basic level of compatibility across audio descriptions and allow one to build new applications. It is this layer that also integrates MPEG-7 Audio with the other parts of the standard. Built upon the foundation layer in the standard are a series of high-level tools that are geared toward particular sets of applications, such as search and retrieval of speech signals. The two most complex high-level tools are detailed in the following chapters and the other high-level tools are outlined here.

17.2 APPLICATIONS

17.2.1 Query by Humming

Consider a streaming audio service, in which there is a database of compressed MPEG-4 audio on one (or more) MPEG-4 media servers and an associated database of MPEG-7 metadata on a MPEG-7 query server. For each MPEG-4 compressed song indexed, the

MPEG-7 database stores a complete representation of the song's melody and some mechanism to link the metadata to the associated media, for example, a URL. The database could also store many other MPEG-7 descriptors, for example, the song title, artist, song length, musical genre and style, musical 'mood' or the instrumental timbre of a characteristic passage.

The user presses the query button on his wireless handheld access device and hums a tune. The sampled signal waveform is transmitted to the query server, where a process is run that extracts its MPEG-7 melody metadata. That metadata is the query target, and the MPEG-7 server searches the MPEG-7 melody database for matches. The top few matches are transmitted back to the access device and displayed as song title and artist. The user can hit a button to immediately start streaming from the MPEG-4 database to the wireless access device, in which the media is decoded and played. The user could follow up the initial query with the more powerful query (e.g. via button and menu selection) to play more songs 'by that artist', or 'in the same genre and style' as that song or just 'with the same mood' as that song.

17.2.2 Query for Spoken Content

Consider a telephone message service with associated MPEG-7 metadata. For each message recorded in a person's message queue, a speech recognition engine processes the voice mail and puts the corresponding text annotation into the MPEG-7 database. The database also stores the calling number (from which some estimation of the caller can be obtained), the time of day of call and the length of the call.

The user is anxiously expecting to receive a specific call. He could call up the message service where the recognition engine could process his voice-based query to form a textual MPEG-7 query. Where the recognition engine fails to convert to text (e.g. because of a proper name being outside of the dictionary), it may use a phonetic representation. The resident MPEG-7 query server could then match the key words in the query with the words in the message annotation and also with the MPEG-7 fields representing the caller. For example: 'Get me all the messages from my boss about the Tallahassee account!'

17.2.3 Extraction and Query Paradigm

The first two examples shared a common architecture. The first step in developing the application is to extract the MPEG-7 metadata from a set of media files, as is shown in Figure 17.1. This could be done once for a largely static set of media files (such as in the query-by-humming example) or done repeatedly (as in the phone message example).

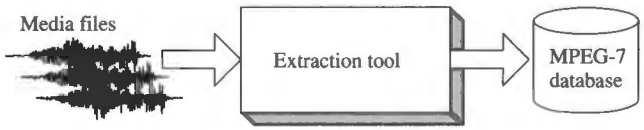


Figure 17.1 MPEG-7 extraction process

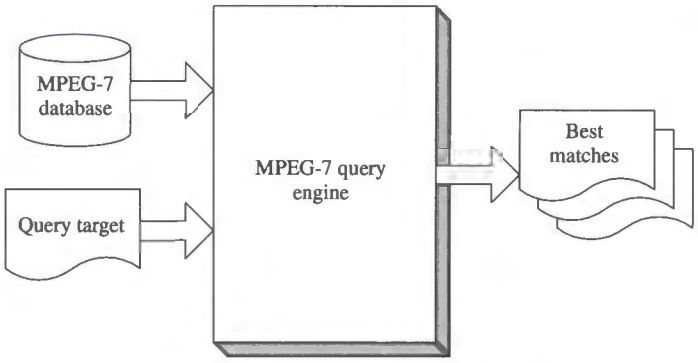


Figure 17.2 MPEG-7 query matching process

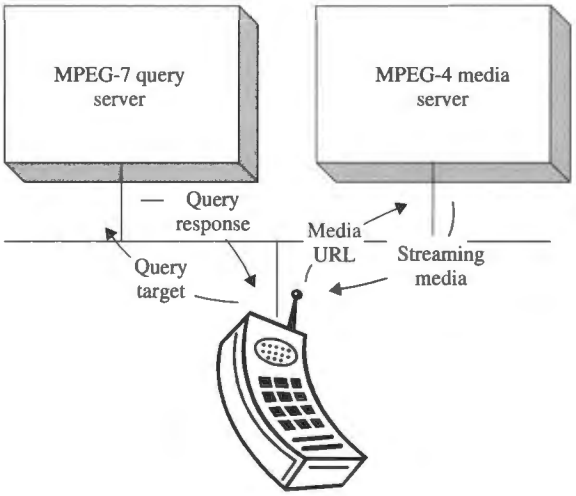


Figure 17.3 MPEG-7 client server architecture

The next step is the query process, as shown in Figure 17.2. In this process, the query is converted to an MPEG-7 query target, typically by the same extraction engine, although this is not shown. The query engine then compares the query target with all the MPEG-7 instantiations in the database via a suitable distance function and returns the best matches. Media-locator information that is part of the metadata can then be used to locate the desired media data. Figure 17.3 shows how this might be combined to form a client server MPEG-7 application, as described in the query-by-humming application.

17.2.4 Assisted Consumer-Level Audio Editing

Imagine that an audio signal has been processed to extract metadata to facilitate interactive applications. For example, many low-level audio features such as power, loudness, spectral centroid and spectral envelope could be used in a hypothetical consumer-level audio editor. One could use these descriptors for display, such as loudness in the place of the waveform.

With some knowledge of standard editing rules within a genre, a sophisticated program could use spectral features to equalize channels and thereby automatically enhance the clarity of a mix.

17.3 OVERVIEW OF AUDIO DESCRIPTORS

The MPEG-7 Audio standard comprises descriptors and description schemes that can be divided into two classes: generic low-level tools and application-specific tools. The generic tools, referred to in the standard as the *audio description framework*, apply to any audio signal and include the scalable series, low-level descriptors (LLDs) and the uniform silence segment. The application-specific tools restrict their application domain as a means to afford more descriptive power and include general sound recognition and indexing tools, instrumental timbre tools, spoken content tools, audio signature tools and melody description tools.

17.3.1 MPEG-7 Audio Description Framework

The audio description framework contains low-level tools that can be used to construct higher level audio applications. By providing a framework for the creation of descriptions and the semantics for commonly used audio features, MPEG-7 Audio establishes a mechanism for the applications to interoperate.

The framework supports two ways of associating low-level audio features with an audio signal. One is to extract features from a series of regular intervals and the other is to extract them from arbitrary Segments (see the Multimedia Description Schemes (MDS) description, particularly Chapters 6 and 8). This association is accomplished via the AudioLLD interface.

17.3.2 The LLD Interface

The low-level audio descriptors have very general applicability in describing audio. There are seventeen temporal and spectral descriptors that can be roughly divided into the following groups:

- *Basic*: Instantaneous waveform and power values;
- *Basic spectral*: Log-frequency power spectrum and spectral features including spectral centroid, spectral spread and spectral flatness;
- *Signal parameters*: Fundamental frequency of quasi-periodic signals and harmonicity of signals;
- *Temporal Timbral*: Log attack time and temporal centroid;
- *Spectral Timbral*: Specialized spectral features in a linear-frequency space, including a spectral centroid, and spectral features specific to the harmonic portions of signals, including harmonic spectral centroid, harmonic spectral deviation, harmonic spectral spread and harmonic spectral variation;

- *Spectral basis representations*: Two features used in conjunction primarily for sound recognition, but generally useful as projections into a low-dimensional space to aid compactness and recognition.

As mentioned above, an LLD may be instantiated either as a single value for a segment or a sampled series. Both of these possibilities are embodied in two LLD types that create a consistent interface. *AudioLLDScalarType* is inherited for scalar values, such as power or fundamental frequency, and *AudioLLDVectorType* is inherited for vector types, such as spectra. Any descriptor inheriting from these types can be instantiated, describing a segment with a single summary value or a series of sampled values, as the application requires.

The sampled values themselves may be further manipulated through another unified interface: they can form a *scalable series*. The scalable series allows one to progressively down-sample the data contained in a series, as the application, bandwidth or storage require. The basic operation of a scalable series is to store all values of a series as vector within a *raw* element. The scalable series may also store various summary values along the way, such as the minimum, maximum, mean and variance of the descriptor values as they are down-sampled (each stored within a *min, max, mean* and *variance* element, respectively). There are many other features within the scalable series, but they are out of the scope of this chapter. The typical use of a series is to be a container for successive sampled values in time or frequency.

The other method of aggregating data for the purpose of description draws heavily (by means of inheritance) from the generic *Segment* multimedia description scheme. For the purpose of MPEG-7 Audio, an audio segment is a temporal interval to which all attached descriptors apply. A segment is a recursive structure, so an audio stream may be hierarchically decomposed [2, 3]. Any audio descriptors or description schemes, including any of the application-based tools and the LLD, may be attached to a given audio segment.

The low-level audio descriptors are all defined within the scope of two sub-types, namely, *AudioLLDScalarType* and *AudioLLDVectorType*. These two types allow for single-value descriptors, such as the power of a waveform's section, and multivalue descriptors, such as spectrum information. Within these two types, the sampling period of a descriptor or, in MPEG-7 parlance, the *hopSize* is defined. The default *hopSize* for all descriptors has been chosen to be 10 ms and descriptors should either utilize the 10-ms period or an integer multiple or divisor thereof to ensure compatibility of descriptors sampled at different rates.

The 10-ms *hopSize* was chosen to maximize compatibility with common audio sampling frequencies. For the vast majority of the commonly utilized audio sampling frequencies, 10-ms sampling results in an integer number of samples per descriptor. In a few rare cases (e.g. 22.05 kHz), 10 ms equates to a noninteger sample number and in such cases there will be minor jitter of the sampling point.

Figure 17.4 gives an outline of a class hierarchy for low-level audio descriptors. The *AudioSegment* illustrated, a descendent of a general *Segment* type, contains an audio descriptor, specifically an *AudioSpectrumCentroid* descriptor, which in turn inherits from an *AudioLLDScalarType*. One single summary value is chosen for the descriptor and is implemented as a *Scalar* (float).

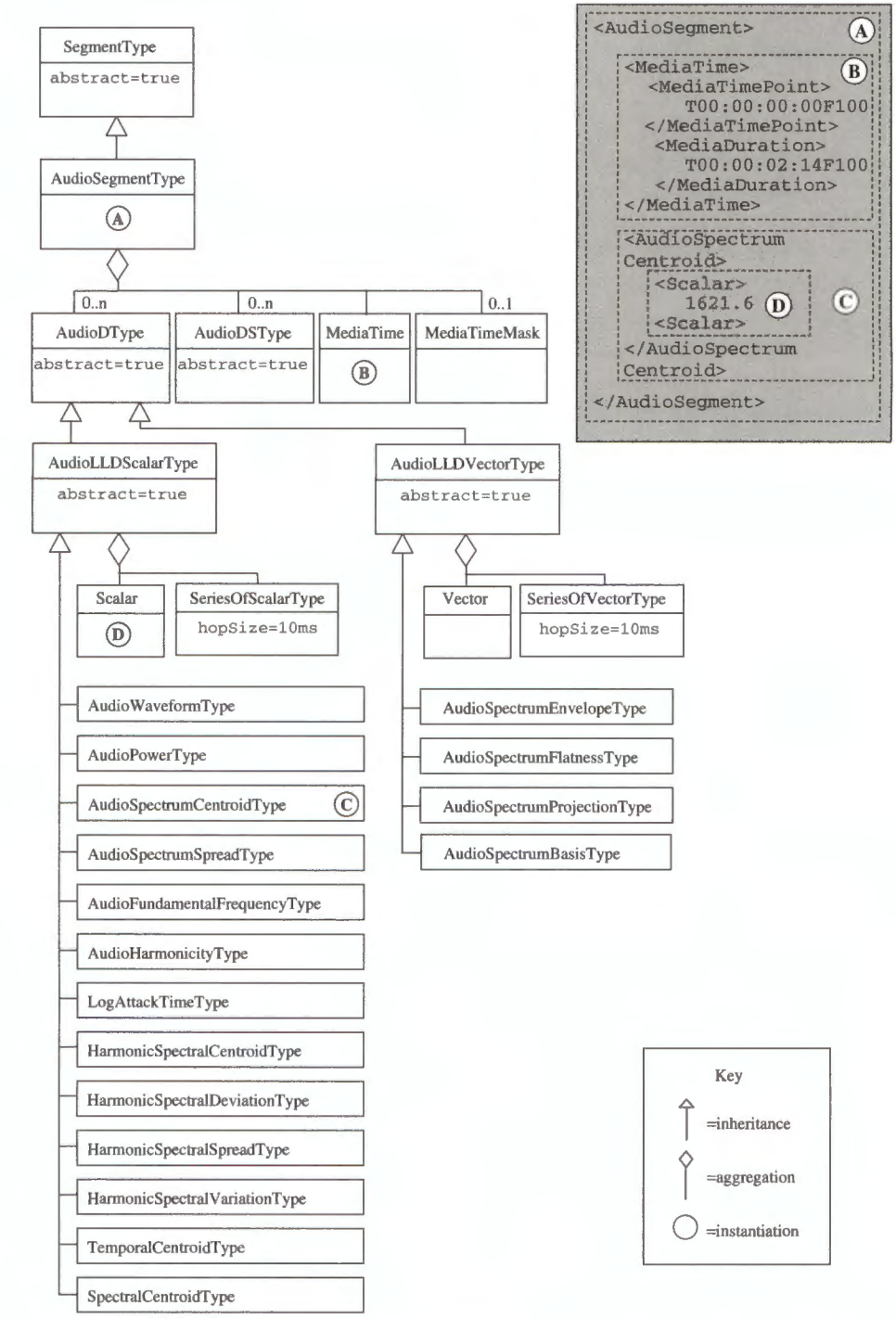


Figure 17.4 Low-level descriptor class hierarchy. The Extensible Markup Language (XML) outlined on the right represents instantiations of the nonabstract types

17.3.3 The Low-Level Audio Descriptors

Basic descriptors

To facilitate our discussions of the basic low-level audio descriptors, we have provided an analysis of a *cor anglais* playing a solo excerpt from the overture *Le Carnaval romain* by Berlioz. This is shown in Figure 17.5, with Figure 17.5(a) showing the original waveform in the time domain.

The two basic descriptors available are the *AudioWaveformType* and the *AudioPowerType*. The first is typically used to describe the minimum and maximum sampled amplitude values reached by the audio signal within the sample period. As an example, this is useful for the display of audio waveforms in a window and is clearly illustrated in Figure 17.5(b) where this descriptor can be seen to give a user a useful model of the envelope of the time domain signal. The second basic descriptor describes the power of the audio signal in the form of the instantaneous power of the samples in the frame. The mean of a 10-ms frame is the fundamental resolution of this descriptor. This gives a temporally smoothed measure of the signal's content as a function of time and for our example is shown in Figure 17.5(c). The power can be used, for example, in conjunction with the other basic spectral descriptors to give economic descriptions of both the time and frequency behavior of the signal. Note that for the purposes of display, the magnitude of the computed power descriptor values have been converted to a logarithmic (decibel) scale.

Basic spectral descriptors

The core of the spectral descriptors is a time-series of logarithmic, sub-band descriptions of the short-term audio signal spectrum. The log-frequency nature of the descriptor serves multiple purposes: firstly, it gives a very compact description of the signal's spectral content, and secondly it mirrors the approximately logarithmic response of the human ear. The logarithmic spectrum is described by the *AudioSpectrumEnvelopeType* descriptor and could be used to generate, for example, a simple spectrogram of the original audio signal. This is the illustrated usage of the *AudioSpectrumEnvelopeType* in Figure 17.5(d). For the purposes of display, the magnitude of the computed descriptor values have been converted to a logarithmic (dB) scale.

The resolution of the *AudioSpectrumEnvelopeType* is controllable between 1/16 of an octave to 8 octaves. For the example, a resolution of 32 bands or 1/4 octave was chosen. The wide resolution range allows a user to choose an appropriate level of spectral descriptor information in terms of the spectral resolution of the logarithmic bands. To ensure that instantiations derived from different signals may be compared, the frequency range of the descriptor is normatively locked. That is to say, the band edges must conform to a simple octave (or power-of-two divisor or multiple thereof) relationship with 1 kHz. The lowest band *loEdge* is normatively 62.5 Hz and the span of the spectrum is then 8 octaves to the *hiEdge* of 16 kHz, which was chosen as a realistic limit of human hearing for this context. Each description vector also contains two extra values representing the energy between 0 Hz and *loEdge* and that between *hiEdge* and the Nyquist frequency of the audio signal. The eight-octave case is a special case as it results in just three coefficients describing the spectrum: within-band, below-band and above-band power.

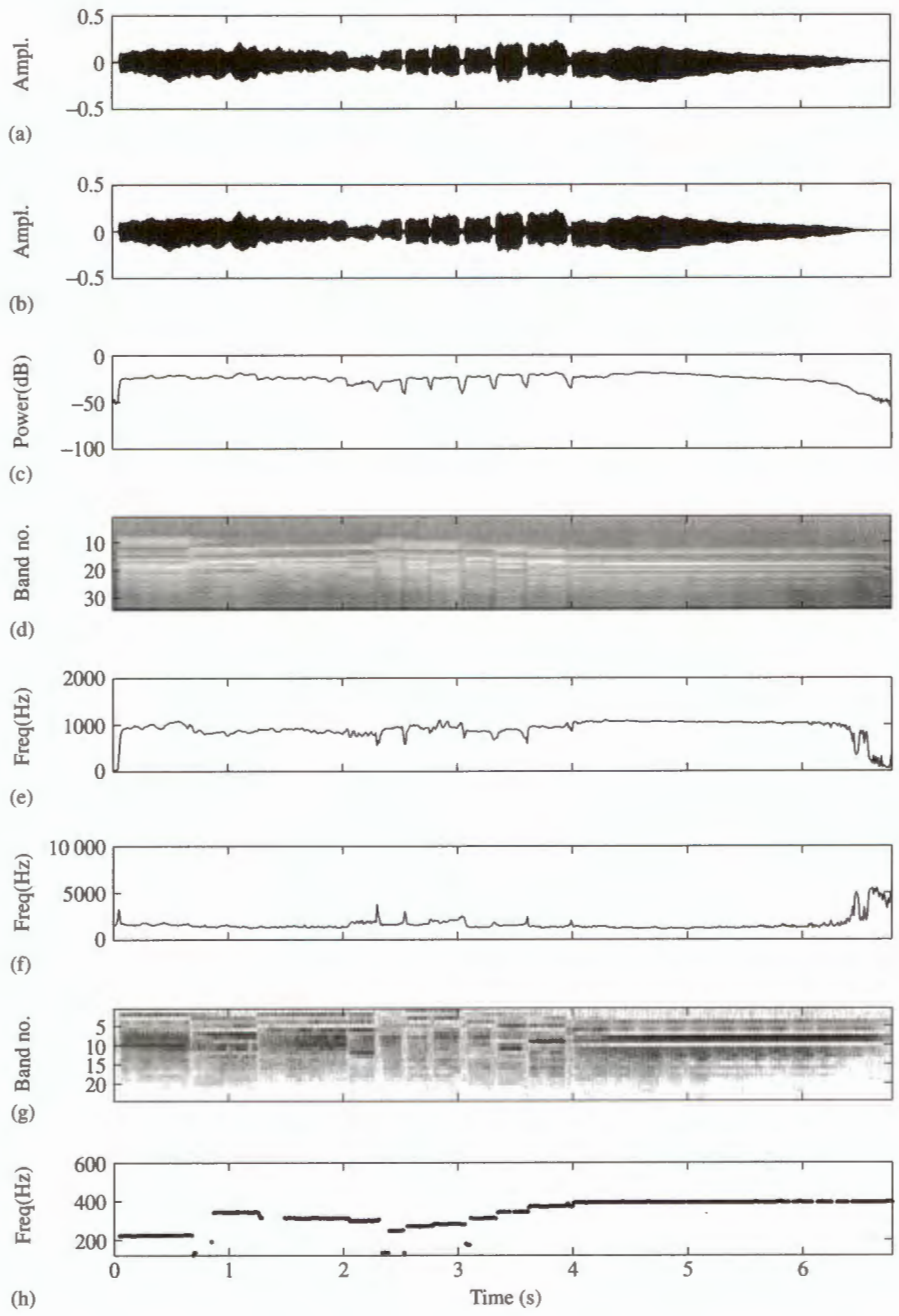


Figure 17.5 *Cor anglais* playing a solo excerpt from the overture *Le Carnaval romain* by Berlioz: (a) Original 44.1 kHz sampled waveform; (b) AudioWaveformType Min-Max representation; (c) AudioPowerType; (d) AudioSpectrumEnvelopeType '1/4 Octave' resolution. (e) AudioSpectrumCentroidType; (f) AudioSpectrumSpreadType; (g) AudioSpectrumFlatnessType 4-band resolution and (h) AudioFundmental FrequencyType

In most cases, the *AudioSpectrumEnvelopeType* can be computed from a simple power spectrum based on the Fast Fourier Transform (FFT) of the frame of audio signal samples. The recommended approach is to use a frame size of 30 ms and a Hamming window function. Zero padding of the frame is used to allow for power-of-two FFT sizes. The mapping between the linear FFT coefficients and the logarithmic frequency bands is performed on an in-band basis with coefficients less than half the FFT resolution from band edges proportionally shared between neighboring bands.

The *AudioSpectrumEnvelope* maintains the power relationship with the original signal preserved in the Discrete Fourier Transform (DFT) through Parseval's theorem. Since the envelope is a power spectrum, the sum of all spectral coefficients is equal to the power in the analysis window.

Complementing the *AudioSpectrumEnvelopeType*, there are a series of spectral feature descriptors that give measures of the signal's spectral content:

- *AudioSpectrumCentroidType*: This describes the center of gravity of the *AudioSpectrumEnvelope*. It is an indicator as to whether the power spectrum is dominated by low or high frequencies and can be regarded as an approximation of the perceptual *sharpness* of the signal. From Figure 17.5(e) it is clear that the *cor anglais* of our example has a spectrum dominated by lower frequencies with a centroid in the 1-kHz range. Here, the displayed values are expressed in Hz rather than the 1 kHz referenced values required in the standard.
- *AudioSpectrumSpreadType*: The second moment of the log-frequency power spectrum described by AudioSpectrumEnvelopeType gives a description of the shape of the power spectrum. In particular, it indicates whether the signal content is spectrally concentrated at the centroid or across a wider range of the spectrum. This gives a mechanism for differentiating noiselike from tonal sounds. In the case of our signal, Figure 17.5(f) suggests that the section of music is relatively consistent without major deviation. This is reasonable for a solo instrument. However, even in this example at the onsets of notes, the deviation can rise markedly. Again, for display purposes, the values of this descriptor are converted to frequency values in Hz.
- *AudioSpectrumFlatnessType*: For this descriptor, the signal is divided into nominal quarter-octave resolution, logarithmically spaced, overlapping frequency bands, and the spectral flatness (ratio of geometric mean to arithmetic mean of spectral power within a band) is calculated for each band. Applications include audio fingerprinting, identification of audio based on a database of known works and, thus, locating metadata for legacy audio content without metadata annotation. A six-octave *AudioSpectrumFlatness* display for the example is given in Figure 17.5(g). In this case, a lighter shade indicates a high value for spectral flatness (which generally means that a tonal component is present or the signal is absent in that band).

Basic signal parameters

The signal parameter LLDs consist of some simple parametric descriptions of the audio signal. However, in practice, the computation of these parameters is subject to estimation. In particular, this grouping includes the computation of the fundamental frequency, F0, of the audio signal. Fundamental frequency computation is generally an inexact science,

but, there are a number of high-accuracy algorithms in the literature [4]. For the purposes of the standard, there is no normative mechanism for F0 computation of the descriptor *AudioFundamentalFrequencyType*; as such, there is scope to use any existing algorithm or future techniques that become available. One normative requirement of all algorithms is to give a measure of periodicity for the signal analysis interval. This is a value between zero and one, in which zero indicates nonperiodic signal content and fills the *weight* field of the *SeriesOfScalars*. This allows the similarity and scaling operations to ignore signal sections that do not contain periodic audio content. The fundamental frequency descriptor gives information regarding musical pitch and the periodic content of speech signals. It can also be used to give indications of melody, but separate melody description schemes are provided for that purpose within the framework. This does not, however, preclude the use of F0 information for melody comparisons.

The second parametric descriptor is *AudioHarmonicityType*. This contains two measures of the harmonic nature of a signal's spectrum. These are *HarmonicRatio* and the *UpperLimitOfHarmonicity*. The *HarmonicRatio* gives a simple measure of the proportion of harmonic components in the power spectrum. The standard avoids dependency on the *AudioFundamentalFrequencyType* computation of F0 by specifying a mechanism for the computation of Harmonic Ratio using the normalized cross-correlation. For the example signal, Figure 17.5(h) gives an illustration of the *AudioFundamentalFrequencyType* evolution with time. In correspondence with the finite resolution nature of the measure, the F0 estimates are plotted at discrete points. For these computations, the F0 track was produced using the normalized cross-correlation technique used by Harmonic Ratio. The *UpperLimitOfHarmonicity* is the point in the spectrum beyond which the spectrum no longer possesses harmonic content. The computation of this point is based on a high- to low-frequency search of the spectra of the original and a comb-filtered version of the signal. The limit is taken as the maximum frequency for which the ratio of the original to comb-filtered power spectra is less than a nominal value of 0.5.

Together, the *AudioFundamentalFrequencyType* and *AudioHarmonicityType* provide a complementary framework to the logarithmic spectral description of *AudioSpectrumEnvelopeType*. These parametric descriptors can describe the 'detail' of the spectrum in terms of the harmonic structure, detail that is necessarily lost in the logarithmic spectral description of *AudioSpectrumEnvelopeType*. Thus, we see that the LLD framework gives an efficient and essential orthogonal and complementary set of spectral descriptors for an audio signal.

Temporal timbral descriptors

The Temporal Timbre descriptors are solely used within an audio segment and are intended to compute parameters of the signal envelope. For this reason, we introduce a second example of a signal segment (Figure 17.6). This is based on an oboe playing a single A = 440 Hz note with vibrato. Notice that the descriptors are strictly limited to the note itself as they have no normative meaning outside of individual sound segments. In particular, the Temporal Timbral descriptors describe the signal's power function over time. The power function is estimated as a local mean square value of the signal amplitude values within a running window. The two descriptors are *LogAttackTimeType* and the *TemporalCentroid*; again, these only make sense within the context of a single, well-segmented sound. The

segment of the computation is limited to the time from the estimated onset to the offset of a perceptually delineated event within the signal. The *LogAttackTime* is defined as the log (base 10) of the time in seconds from a signal's starting point to the time that it reaches a sustained section (for a harmonic signal) or a maximum (for a percussive sound). For the example shown in Figure 17.6(a), the *LogAttackTime* was computed as $-1.44\log_{10}(s)$, (0.0363 seconds). The *TemporalCentroid* computes a time-based centroid of the signal envelope for the signal segment and for the example it can be computed as 0.9561 seconds. Both these descriptors have well-defined roles within the high-level tools provided for timbre; however, the inclusion of these descriptors as LLDs opens their use to further and more varied applications.

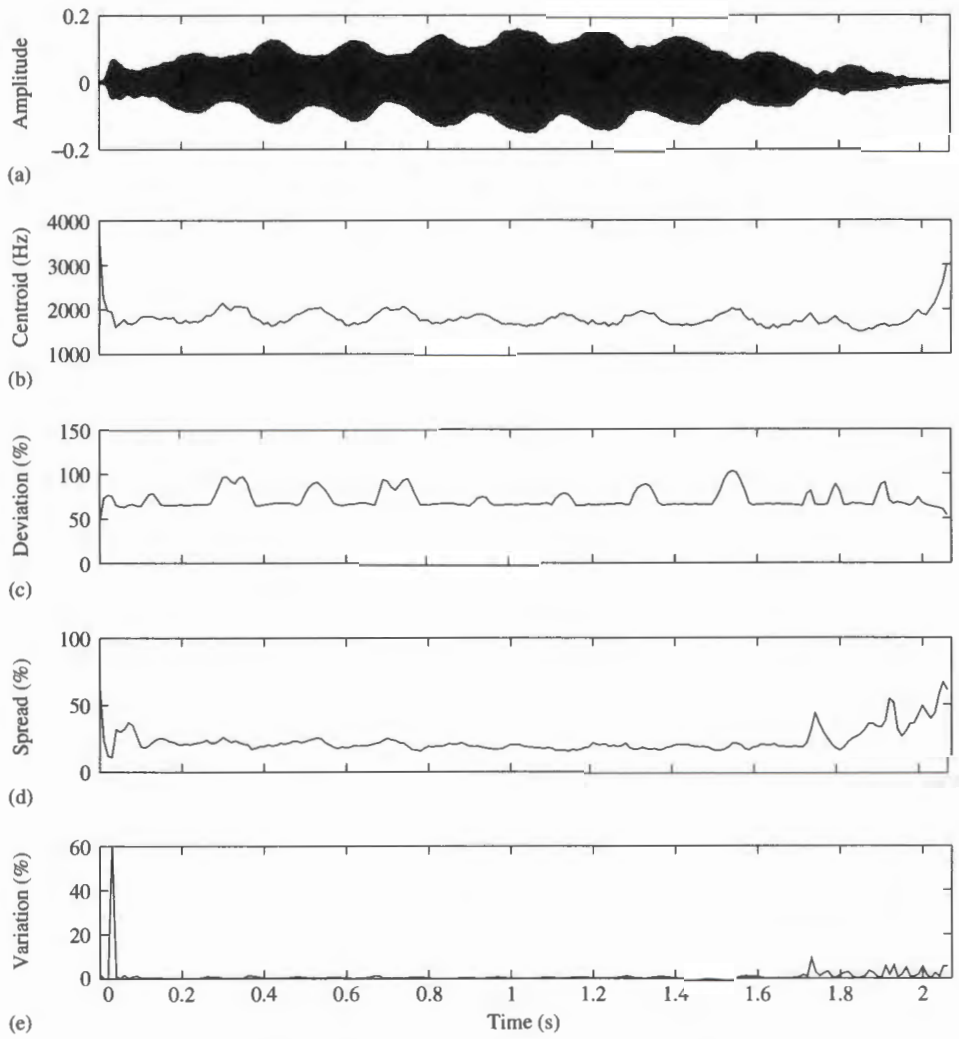


Figure 17.6 Oboe playing A = 440 Hz with vibrato: (a) Original 44.1 kHz sampled waveform; (b) HarmonicSpectralCentroidType; (c) HarmonicSpectralDeviationType; (d) HarmonicSpectral SpreadType and (e) HarmonicSpectralVariationType

Spectral timbral descriptors

The timbre descriptors were designed to be computed with a frame size that is a whole number of fundamental periods in length. Many of the timbre descriptors can also be used for wider signal characterization purposes. Thus, the standard carries informative suggestions of either a Hamming window of eight fundamental periods with a hop size of four periods, which aids in spectral peak picking, or a Hamming window of 30-ms duration with a hop size of 10 ms, which fits in the LLD framework, for the computation of the Spectral Timbral descriptors. In Figure 17.6, the 30-ms window, 10-ms *hopSize*-derived descriptors, are shown.

The Spectral Timbral descriptors contrast with the main set of basic LLDs in that they are computed in a linear-frequency space. As such they complement the Basic descriptors and expand the possibilities offered by the two Basic Signal Parameter descriptors. The set of descriptors contains *HarmonicSpectralCentroid*, *HarmonicSpectralDeviation*, *HarmonicSpectralSpread*, *HarmonicSpectralVariation* and *SpectralCentroid*. The first four of these are given for our vibrato oboe signal in Figures 17.6(b) to 6(e), respectively. It is clear that the *HarmonicSpectralCentroid* is relatively stable with some minor variation caused by the vibrato in the note. The *HarmonicSpectralDeviation* gives a measure of the spectral deviation from a local spectral envelope; Figure 17.6(c) clearly illustrates the spectral variation within the vibrato note, showing the added modulation. The *HarmonicSpectralSpread* of the signal gives the power-weighted RMS deviation from the *HarmonicSpectralCentroid*; hence, Figure 17.6(d) indicates the vibrato modulation but is not as obvious as the *HarmonicSpectralDeviation*. The *HarmonicSpectralVariation* gives the spectral variation between adjacent frames, and since the vibrato is a slowly varying modulation, the *HarmonicSpectralVariation* is quite low for the majority of the note. The latter would be expected for a single note from a single woodwind instrument, except at the release of the note where the variation can be seen to rise significantly as sudden temporal changes occur and noise dominates as the signal power decreases.

It is also interesting to consider the mean values of the Harmonic spectral descriptors. For the signal shown in Figure 17.6, these are displayed in the first column of Table 17.1, while the second column are the results of the same note played on the same *cor anglais* for similar duration but without vibrato. This illustrates the use of the descriptors in a general discrimination of a sound signal. In this case it is noticeable that the vibrato and nonvibrato notes differ mainly in the value of the *HarmonicSpectralVariation* descriptor.

Table 17.1 Comparison of mean Harmonic descriptor values for an oboe playing the note A4 with and without vibrato

Descriptor	With vibrato	Without vibrato
HarmonicSpectralCentroid (Hz)	1803.5	1820.2
HarmonicSpectralDeviation (%)	70.95	73.87
HarmonicSpectralSpread (%%)	22.74	25.01
HarmonicSpectralVariation (%)	0.96	0.55

Spectral basis representations

The final section of the LLD represents some novel ideas that have appeared in the audio signal processing literature in recent years [5, 6]. In particular, the spectral basis representations offer a tool to project an audio signal's spectrum (a high-dimensional representation) into a low-dimensional representation. The descriptors embody the basis functions themselves (*AudioSpectrumBasisType*) and the projection of the spectrum using the basis function (*AudioSpectrumProjectionType*). The standard describes normative techniques using singular value decomposition (SVD) and Independent Component Analysis (ICA) methods. An example of the power of the techniques is demonstrated by the reconstruction of a full-bandwidth spectrogram from just ten basis components containing just one-sixth of the information required for the original. These ideas are used further in the General Sound Recognition and Indexing section of the audio standard, which is described in detail in Chapter 19.

Silence segment

A very simple but useful tool is the MPEG-7 silence descriptor. It attaches the simple semantic of 'silence' (i.e. no significant sound) to an audio segment. It also includes a simple indicator of the level of silence via the minimal temporal threshold. It may be used to aid further segmentation of the audio stream or as a hint not to process a segment.

17.3.4 High-Level Description Tools

General sound recognition and indexing tools

The general sound recognition and indexing descriptors and description schemes are a collection of tools for indexing and categorization of sound effects and sounds in general. Support for automatic sound effect identification and indexing is included, as well as tools for specifying a taxonomy of sound classes and tools for specifying an ontology of sound recognizers. Such recognizers may be used to automatically index and segment sound tracks. These tools are detailed further in Chapter 19.

Spoken content description tools

The Spoken Content description scheme is predicated on the idea that speech recognition systems are currently imperfect. Rather than being a simple textual transcript of what is spoken (although it can accommodate such a thing), the description scheme consists of combined word and phone lattices for each speaker in an audio stream. By combining the lattices, the problem of out-of-vocabulary (OOV) words is greatly alleviated and retrieval may still be carried out when the original decoding was in error. The description scheme can be used for two broad classes of retrieval scenario: indexing into and retrieval of an audio stream and indexing of multimedia objects annotated with speech. The Spoken Content tools are detailed in Chapter 18.

Musical instrument timbre description tools

Timbre Descriptors aim at describing perceptual features of instrument sounds. Timbre is typically defined in the literature [7] as the perceptual features that make two sounds having the same pitch and loudness sound different. The Timbre description scheme describes these perceptual features with a reduced set of descriptors. The descriptors relate to notions such as 'attack', 'brightness' or 'richness' of a sound.

There are four classes of musical instrument sounds: harmonic, sustained, coherent sounds; nonharmonic, sustained, coherent sounds; percussive, nonsustained sounds and noncoherent, sustained sounds [8]. Within these four classes, two classes are well detailed within the MPEG-7 standard and have been the subject of development. They are the harmonic, coherent, sustained sounds and nonsustained, percussive sounds. The other two classes were deemed to be of lower priority because of their relative rarity, but may still make an appearance in the standard. As shown in Table 17.2, the timbre description scheme for sustained harmonic sounds combines the above Timbral Spectral LLDs with a log attack time descriptor. The percussive instrument description scheme combines the Timbral Temporal LLDs with a spectral centroid descriptor. Comparisons between descriptions using either set of descriptors are done with an experimentally derived scaled distance metric.

Melody description tools

The Melody description tools include a rich representation for monophonic melodic information to facilitate efficient, robust and expressive melodic similarity matching. The Melody Description Scheme includes a Melody Contour Description Scheme for extremely terse, efficient melody contour representation and a Melody Sequence Description Scheme for a more verbose, complete, expressive melody representation. Both tools support matching between melodies. One may also take advantage of additional, optional support descriptors such as lyrics, key, meter and starting note, in order to constrain search in applications such as query by humming.

The Melody Contour Description Scheme uses a five-step contour (representing the interval difference between adjacent notes), in which intervals are quantized. The Melody Contour Description Scheme also represents basic rhythmic information by storing the number of the nearest whole-beat of each note, which can dramatically increase the accuracy of matches to a query.

Table 17.2 The low-level descriptors contained in the two Timbre description schemes

Harmonic	Percussive
HarmonicSpectralCentroid	LogAttackTime
HarmonicSpectralDeviation	TemporalCentroid
HarmonicSpectralSpread	SpectralCentroid
HarmonicSpectralVariation	
LogAttackTime	

For applications requiring greater descriptive precision or reconstruction of a given melody, the Melody Sequence Description Scheme supports a high precision of interval encoding. Rather than quantizing to one of the five levels, the precise pitch interval (to cent or greater precision) between notes is kept. Precise rhythmic information is kept by encoding the logarithmic ratio of differences between the onsets of notes in a manner similar to the pitch interval.

17.3.5 Other Parts of the Standard

The audio part of the standard was designed to work well with the other parts of MPEG-7, and is especially tied to Part 5–MDS. To avoid redundant functionality within the standard, the audio subgroup relies on the efforts of other groups for completing the set of tools that an audio archivist might need. We have already discussed the use of the DDL as a standard way of extending the standard. We have also seen how the audio framework utilizes the general segment from the MDS.

Anyone describing content may wish to attach typical descriptors such as title, composer and year of recording to a piece of music. These descriptors, which are in current practice in libraries and archives, are covered in detail in the MDS, particularly in the section on creation and production information. Similarly, one may try to describe musical genre as a hierarchical taxonomy or identify a musical instrument from a list of controlled terms. The mechanisms by which one can create these ontologies and dictionaries are in the MDS as the Controlled Term data type and the classification scheme description scheme. One describes aspects of the medium itself, such as the encoding format or sound quality, with the media description tools in the MDS. By putting all of these parts together, one can transfer an existing archive to a unified framework under the MPEG-7 standard, and then enhance it further with specialized, signal processing-derived audio description schemes. For further details, see Section III – Desctiption Schemes.

17.4 SUMMARY

In this chapter, we have given an overview of the MPEG-7 Audio standard. In it, we discussed the various applications that might apply to audio description and how the standard is organized into high- and low-level tools. In the overview of the audio descriptors, we outlined the various low-level signal-processing features that are supported directly in the standard and how to use those descriptors to describe the structure of an audio document. We also gave a brief summary of the available high-level tools and how to relate high-level audio archive descriptions to capabilities described in other parts of the standard. In the two following chapters, the most complex but wide-ranging audio description tools, on spoken content and sound recognition, are detailed.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Rosemary A. Fitzgerald for her assistance in preparing this chapter.

REFERENCES

[1] ISO/IEC FDIS 15938-4, Information Technology – Multimedia Content Description Interface – Part 4 Audio. Geneva, (in preparation) 2002.

[2] A. T. Lindsay and W. Kriechbaum, "There's more than one way to hear it: multiple representations of music in MPEG-7" *Journal of New Music Research*, **28**(4), 364–372 (1999).

[3] A. T. Lindsay, S. Srinivasan, J. P. A. Charlesworth, P. N. Garner and W. Kriechbaum "Representation and linking mechanisms for audio in MPEG-7", *Signal Processing: Image Communication*, Vol. 16, 2000, pp. 193–209.

[4] W. Hess *Pitch Determination of Speech Signals: Algorithms and Devices*, Springer-Verlag, Berlin, New York, 1983.

[5] M. Casey, Auditory Group Theory: with Applications to Statistical Basis Methods for Structured Audio, Ph.D. Thesis, MIT Media Lab, Cambridge, Mass., February 1998.

[6] M. A. Casey and A. Westner, Separation of mixed audio sources by independent subspace analysis, Proceedings of the ICMC 2000, International Computer Music Conference, Berlin 27 August–1 September 2000.

[7] American National Standards Institute (ANSI), *Psychoacoustical Terminology*, ANSI S3.20, American National Standards Institute, New York, 1973, p. 56.

[8] G. Peeters, S. McAdams and P. Herrera, "Instrument sound description in the context of MPEG-7", Proceedings of the ICMC 2000, International Computer Music Conference, 2000, Berlin, 27 August–1 September.

18

Spoken Content

J. P. A. Charlesworth[1] and Philip N. Garner[2]

[1]Reech Capital PLC, London, United Kingdom, [2]Canon Research Centre Europe Ltd, Berkshire, United Kingdom

The words spoken in an audiovisual document form an obvious and intuitive metadata component. This component is essential to ensure comprehensive coverage of audiovisual content by the MPEG-7 standard. With manual transcription prohibitively costly, such metadata will typically be derived from automatic speech recognition (ASR) systems. The errors inherent in the output of such extraction tools cause particular difficulties for robust retrieval, as well as for interoperability in heterogeneous databases. We describe the SpokenContentDescriptionScheme – a structure comprising a probabilistic combined word and phone lattice along with an explanatory metadata header – and detail how this structure avoids or ameliorates these problems.

18.1 SPOKEN CONTENT IN THE CONTEXT OF MPEG-7

18.1.1 Positioning Spoken Content Within MPEG-7

The MPEG-7 metadata standard encompasses a wide range of descriptors. At one extreme are the low-level descriptors (LLD), for example, color histogram, in which a mathematical transformation is performed on the audiovisual document. Such metadata is readily extractable in software or hardware but is seldom easily interpretable in human terms. In some cases, this metadata may only be utilized in conjunction with other low-level metadata. At the other extreme, MPEG-7 defines high-level description schemes containing metadata such as a person's name or title. Except in rare cases, in which these may be extracted from auxiliary metadata (such as teletype captioning), the contents of these description schemes must be manually created. Naturally, high-level metadata has the benefit of immediate human comprehension. The position on this spectrum occupied by a descriptor or description scheme is largely defined through its extraction method and human comprehensibility. Metadata components that fall in the middle of this spectrum (i.e. where machine derived metadata is used to approximate high-level metadata) prove

particularly problematical, being able to resort to neither mathematical rigor nor human comprehensibility to ensure their usability.

The spoken content is one example of data yielding such problematical metadata, and is represented in MPEG-7 by the Spoken Content set of description schemes (referred to hereafter as the SpokenContentDescriptionScheme). For such metadata, the distinction between a descriptor and a description scheme is unclear. In the case of spoken content, the lattice has the form of a descriptor as it is automatically extracted. It is referred to as a description scheme, however, as it is both able to be referenced as an entity in itself, and contains other MPEG-7 entities such as time.

Perhaps one of the most obvious forms of metadata extractable from an audiovisual document, the SpokenContentDescriptionScheme represents the actual words spoken. Being the primary means of interpersonal communication, a significant percentage of the useable information may reside solely in the spoken content. In audiovisual material, while descriptors such as key frames provide a visual representation of action development, the understanding of the content most readily comes from words spoken. In extreme cases such as voice mail, there is little useable content other than speech.

18.1.2 Importance of Spoken Content

Aside from annotating existing audiovisual documents, the existence of a spoken content description scheme enables spoken annotation where other interfaces would be impractical. For example, most digital camera users are unlikely textually to annotate photographs at a later stage and, because of restrictive interface size, are unable to do so at the time of taking the photograph. Voice annotation of photographs at the time of taking may prove the only way of organizing the large number of (essentially) free photographs possible from digital cameras.

The importance of the spoken content, however, lies not simply in its prevalence but also in its direct and natural usability. Although spoken words and textual words are different from the algorithmic standpoint, humans naturally switch between the two. Thus, querying a spoken document by entry of textual words is perceived as a perfectly natural way of retrieving an audio segment, and fits with the user's mental model.

Were the SpokenContentDescriptionScheme derived from manual transcription, we would expect to be able to leverage the mature technologies of standard textual retrieval. Evidently, in the case of this ideal spoken content metadata, we would categorize it as high-level metadata. Unfortunately, hand transcription of spoken content metadata is either impractical or uneconomic, except in rare instances in which scripted speech takes place, for example, a news broadcast or political speech.

In principle, retrieval of words spoken is possible using query-by-example (QBE) employing low-level audio descriptors such as 'spectrum' or 'energy'. In practice, however, such methods are incapable of scaling to large database sizes. Consequently, our only useable interface to spoken content is through the SpokenContentDescriptionScheme. The SpokenContentDescriptionScheme has been designed to accommodate the output of large vocabulary ASR. Such tools are now readily available from several different commercial organizations, either off the shelf or as part of customized packages.

18.2 ASR TECHNOLOGY TODAY

18.2.1 Drawbacks of ASR

The utility of ASR as an extraction tool in the creation of MPEG-7 metadata raises a number of issues. Extraction tools such as ASR are considered nonnormative. A typical ASR system may contain several million parameters adapted to user and environment. Consequently, the output of any two ASR systems can differ significantly, making retrieval from heterogeneous spoken content databases problematical. In addition, the metadata must be robust to decoding errors to ensure that annotated data remains retrievable; a user will consider it unacceptable that a clearly enunciated spoken phrase, which they know to reside in the audio, cannot be recovered from the spoken content metadata. Thus, the use of ASR for spoken content metadata extraction creates problems in both interoperability and error tolerance.

18.2.2 Applications

The SpokenContentDescriptionScheme has been designed with two distinct types of application in mind:

1. Annotation of media where the spoken content forms an intrinsic part of the media itself. Typically, this could be a television broadcast featuring a journalist interviewing a politician. There may be more than one participant in the spoken content, and the content may or may not be scripted.
2. Annotation of media where the spoken content is itself an annotation of that media. An example of this is where a user takes a photograph using a digital camera and speaks an annotation. The spoken annotation may consist of the names of the people or places in the photograph.

Because of this ambiguous meaning of the word 'annotation', we refer to a spoken content lattice as a transcription.

The two applications scenarios represent different constraints and design goals. In the first case, the transcription is likely to be several minutes long, perhaps even hours. The audio quality may vary throughout the transcription. In the second case, the transcriptions are likely to be very short – less than 10 seconds – but have a consistent audio quality. We may, additionally, propose many other scenarios, for example, telephone conversations whose complexity lies somewhere along this axis.

It follows that the class of ASR used for each application will be different. The first case may be a sophisticated recognizer with a vocabulary of several thousand words running on a platform of considerable computing power. The second case is likely to concern an embedded device with a smaller vocabulary and higher probability of error.

18.3 METADATA ERRORS IN EXTRACTION – A PHYSICAL ARGUMENT

To understand the design restrictions on the spokenContentDescriptionScheme, it is essential first to understand the nature of the extraction tools and their failure modes. A



Figure 18.1 Extraction pathway converting audio data into spoken content metadata

typical extraction path is illustrated in Figure 18.1. The output of an ASR system may be considered a stochastic corruption of the idealized hand transcription. Word accuracy may vary from over 90% for read speech using speaker adapted (or speaker dependent) systems, through 70% for real-world speech and down to as low as 30% for conversational speech [1]. Term accuracy, that is, the accuracy of high-information words such as nouns is generally significantly lower because of the larger range of potential terms.

As with any statistical pattern matching method, errors and misclassifications occur. The causes and nature of the errors may be loosely divided into three classes described below. Although the occurrence of different types of error will vary according to the usage of the SpokenContentDescriptionScheme, MPEG-7 is designed to represent metadata for the full range of audiovisual applications, and consequently must be as robust as possible to all these errors.

18.3.1 Quality of Audio

The first category of error is, perhaps, the easiest to understand. As the distortion on the audio signal increases, for example, because of background noise or channel corruption, the capability of the ASR to identify discriminative components of words diminishes. Words that differ phonetically only slightly, for example, the quiet /f/ phoneme distinguishing 'scar' and 'scarf' or the similar /jh/ and /ch/ phonemes in 'Jews' and 'choose' can very readily be confused. As the noise or corruption level increases, a wider range of errors becomes possible. Under some circumstances audio preprocessing can reduce (but seldom entirely remove) these effects. The duration of the speech can have additional detrimental effects on accuracy due to contextual problems. Previous experiments have largely centered on the TREC [3] series of evaluations using long annotations and large numbers of query words. As annotation length decreases, the decoding accuracy also decreases. This will be significant in cases of annotation of photographs, where short captions are likely. The effect of such misrecognitions may be significantly reduced via the retention of multiple hypotheses within the extracted metadata, although at a cost to precision [4].

18.3.2 Quality of Speech

The second category of error can yield a very significant speaker and context dependent variation in accuracy. ASR systems typically employ phone lexicons[1] representing the canonical pronunciation of words. Accents and dialects other than those for which the

[1] Phones differ from phonemes in that while phonemes are phonologically derived, phones are data derived. Consequently, ASR systems employ phones and we will use this term throughout this chapter.

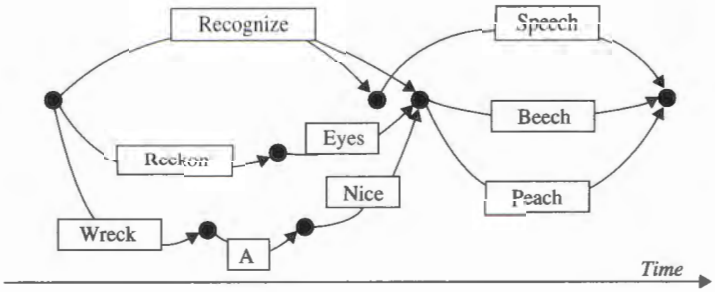


Figure 18.2 Hypothetical decoding of the phrase 'recognize speech' showing the range of ambiguities and misrecognitions

system was trained can result in suboptimal matching of phones yielding misrecognition of words. Similarly, nongrammatical speech, elisions and hesitations can further lower the recognition rate. This effect is especially significant for real-world speech.

18.3.3 Quantity of Training Data

Lastly, and of especial importance for retrieval, we must consider errors resulting from words outside the ASR training set; so-called out-of-vocabulary (OOV) errors. The word in these cases, for example, a person's name or a place, is unknown to the ASR system; one or more other words are substituted in its place resulting in potential corruption of adjoining words. As nouns form the majority of retrieval terms, misrecognition of such a word can yield unrecoverable annotations.

The effects of the errors from the second and third categories can be reduced through the incorporation of phonetic decodings [4–6]. Where an entire word may be decoded incorrectly, just a few of the corresponding decoded phones may be in error. Although semantic searching will not be possible directly,[2] expression-based audio retrieval will be possible through a search for sections of the annotation that 'sound like' the query. It should be noted that it is up to the ASR system whether to provide words, phones or even both [8].

In summary, we must assume that the extraction tool will create imperfect metadata. To ensure retrieval where the best-path word decoding is in error we may include further word hypotheses in the form of a lattice. In addition, we may also include phonetic decoding to support expression-based retrieval to ensure that OOV terms are retrievable. The breadth of the lattice and ratio of word to phone content may be varied according to confidence in the ASR decoding as well as storage capacity. A portion of a typical word lattice is illustrated in Figure 18.2.

18.4 INTEROPERABILITY ACROSS EXTRACTION TOOLS – A TECHNOLOGICAL ARGUMENT

Large Vocabulary ASR is available in many forms from many different suppliers. The underlying technology of all these options, however, is broadly the same. Even putting

[2] Semantic searching is possible indirectly either through a canonical phone dictionary or through a text-to-phone algorithm.

aside the Hidden Markov Model (HMM) technology on which most ASR systems are currently based, all systems use word models made up of some phonologically inspired subword units. The task of speech recognition becomes a task of arranging the known subword units into a word or sequence of words that best explain the spoken content being decoded. In the particular case of HMM systems, a process known as dynamic programming produces a ranked set of possible explanations. In the case of a dictation system, the ASR discards all but the best explanation yielding a single string of words. A more advanced dictation product might keep the different explanations in order to present them to a user in the form of word alternatives (to enable error correction). Postprocessors are often utilized to output N-Best parses or word alternatives, depending on the application. In general, an ASR system produces a lattice of words, identical to that in Figure 18.2 as a natural result of the underlying decoding technology. In other words, the technology of ASR naturally leads to a representation that can also be justified from a linguistic or noisy channel point of view.

The lattice also functions as a convenient superset of commercial ASR output capabilities. A simple text string can be represented trivially as a lattice in which each node gives rise to a single link. The case in which an ASR engine provides word alternatives can also be represented as a lattice in which word alternatives begin at the same node and end at 'sink' nodes, as in Figure 18.3. Note that this arrangement is not strictly a lattice in the sense that an HMM based ASR system would produce, but does represent the word alternative information. In particular, the 'alternative' words do not necessarily begin and end at the exact times specified by the lattice. If an application chooses to use the SpokenContentDescriptionScheme in this manner then the designer should be sure that the timing error introduced is not significant in the context of the application.

The design of the SpokenContentDescriptionScheme is based on the premise that a simple, ASR-derived, textual string representing the spoken content is insufficient. Although most commercial ASR systems produce a textual string at the output, internally they store alternatives and statistics about those alternatives. The SpokenContentDescriptionScheme is capable of storing this range of considered hypotheses, as well as a quantification of the errors.

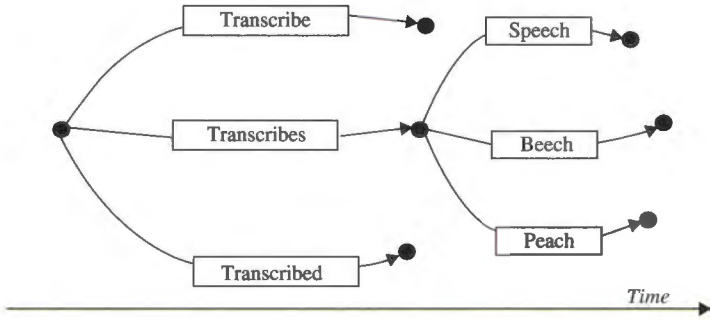


Figure 18.3 A lattice representing word alternatives for the expression 'transcribed speech'. Note that hypotheses start at the same time although they may terminate at different times

18.5 INTEROPERABILITY ACROSS DATABASES

We have demonstrated that a fundamental consideration in the design of the SpokenContentDescriptionScheme is the ability to support the (potentially) erroneous decodings extracted using an ASR system in a manner amenable to robust retrieval. We must now consider how we may utilize a heterogeneous spoken content database, that is, a database consisting of spoken content metadata extracted using a number of different ASR engines. As discussed above, the complexity, size and cost of ASR systems, as well as specialization through speaker and dialect adaptation, and their continuing development, means that the ASR system cannot be made normative. Unfortunately, the output of different ASR systems can differ markedly.

The problem faced is that of how to ensure that retrieval rankings for two (or more) annotations are placed on an even footing when their error statistics differ. A particular decoding may be ascribed a low likelihood, either because it is an improbable path through the lattice of hypotheses or because the decoder has a high probability of error. Less trust should be placed on low confidence metadata than higher confidence metadata. To support fully statistical retrieval, the corruption channel applicable to word decoding would need to be completely characterized through long context conditional confusion probabilities. This is impractical, both from memory considerations as well as because of the paucity of such data. Instead, three levels of information are supplied to assist in retrieval.

The most important is the existence of probability values on word (and, indeed, phone) links (see Figure 18.4). These give an indication of the confidence of the different words starting at that time. These probabilities can be approximated by the confidence information supplied by ASR systems.

The next level is the size of the lexicon used in the decoding (which may be significantly larger than the range of words in the annotation); where a word is decoded from a small, highly constrained, range of possibilities one need to place less reliance

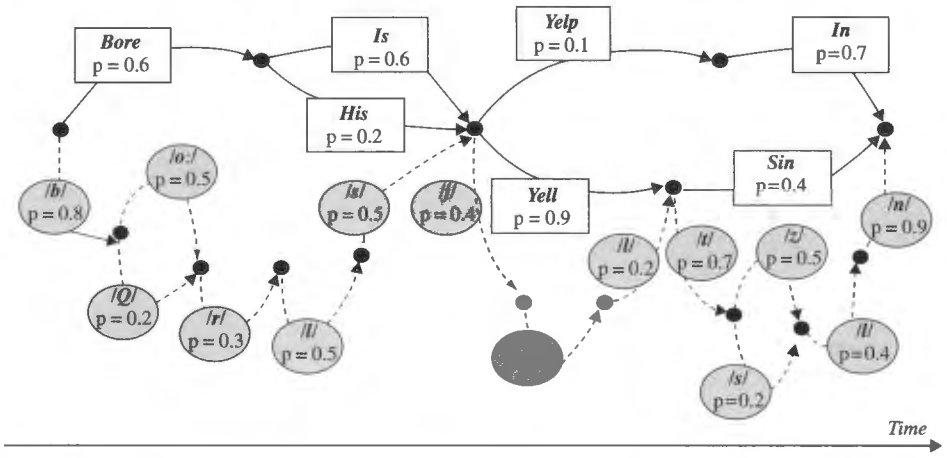


Figure 18.4 Probabilistic combined word and phone lattice representing the OOV words 'Boris Yeltsin'

on it than where the range is large. In many ASR systems, any noise or speech will be decoded into one of the lexicon words irrespective of likelihood.

A final assistance is a coarsely enumerated type to indicate the quality of the audio, whether it is noise, noisy speech or clean speech. Although a rough, discrete measure such as this is far from ideal, the issue of quantifying the noise and the impact on decodings is not a clear-cut issue and this limited quantization provides the most realistic measure.

For phones, we may provide a wider range of information. The number of phones in a language is several orders of magnitude smaller than the number of words; while the commonly used vocabulary of an English speaker may number $O(10^5)$ words, the number of phones is ~45. It thus becomes possible to include not just the instantaneous likelihood of a phone within the lattice but additionally the statistics of phone insertions, omissions and mutual confusabilities. In this way, we ensure that systems of strongly differing quality may be treated equally.

18.6 STRUCTURAL OVERVIEW OF THE SPOKENCONTENTDESCRIPTIONSCHEME

Although the components required by error robustness and interoperability provides a minimum useable structure, practical considerations imposed by present and future usage as well as hardware and software platforms must also be incorporated into any design.

Consider, for example, an individual taking a collection of photographs and, simultaneously, annotating these still images with spoken captions. It is reasonable for the content metadata, that is, the combined word and phone lattice, to be linked to the photograph. The location of the explanatory metadata,[3] that is, the lexicons and phone statistics is, however, less clear. In particular, where one speaker has used one ASR and one language for all annotations, the storage overhead of including explanatory metadata with each photograph becomes unreasonable. Rather we should make use of a single repository for explanatory metadata and all lattices that make use of this should reference it. For the photo annotation scenario, we thus have the structure illustrated in Figure 18.5. The SpokenContentDescriptionScheme is thus composed of two distinct but interrelated description schemes: the SpokenContentHeader and the SpokenContentLattice with references between them [9].

18.6.1 SpokenContent Header

As detailed above, robustness and interoperability considerations result in a structure consisting of two separate but linked components: a header, containing explanatory metadata, and lattices, containing a probabilistic lattice representing the words and phones spoken. In this section, we will describe the detailed structure and utility of the elements in the header. The structure of the header is illustrated diagrammatically in Figure 18.6.

The header contains four groups of explanatory metadata describing who is speaking, what words they are speaking, what phone set their speech is represented in and the

[3] Sometimes referred to as meta-metadata.

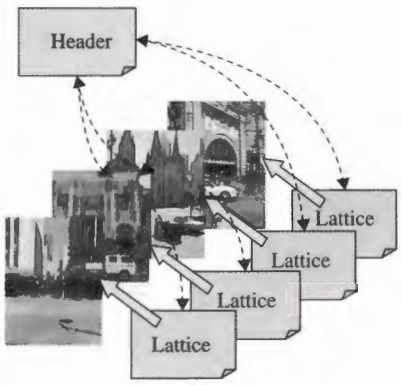


Figure 18.5 Lattices may be used to annotate other metadata. The lattices link to the header that contains the relevant explanatory metadata

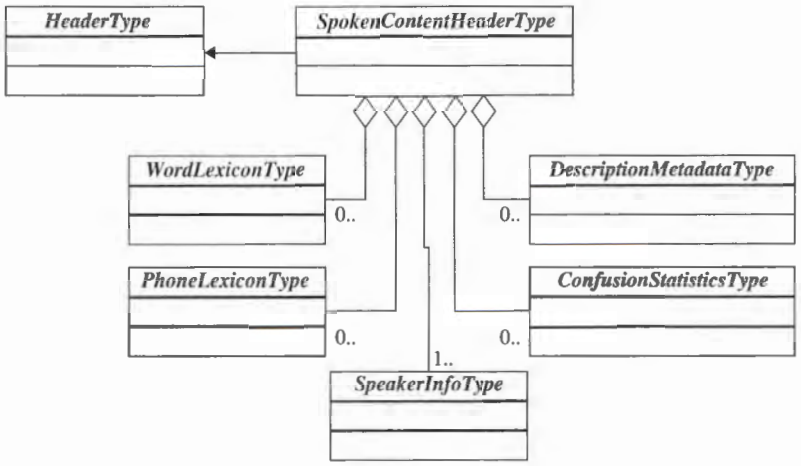


Figure 18.6 The header contains static information that may apply to more than one spoken content transcription

statistical properties of the decoding errors of the phonetic speech recognition system. By combining these elements we may describe the speech of many speakers in a compact form by reusing common components produced by, for example, using a single ASR system or vocabulary for multiple speakers' annotations.

18.6.2 Word Lexicon

The purpose of the word lexicon is to store in a single location the entire utilized vocabulary of one or more ASR systems. In practice, it may additionally be useful during retrieval as it contains information on the size of the word hypothesis space employed during decoding. The ASR lexicon of a large vocabulary speech recognition system can consist of more than 100 000 words while the vocabulary range of an hour's normal speech may be just a few thousand words. For memory efficiency one will often just list

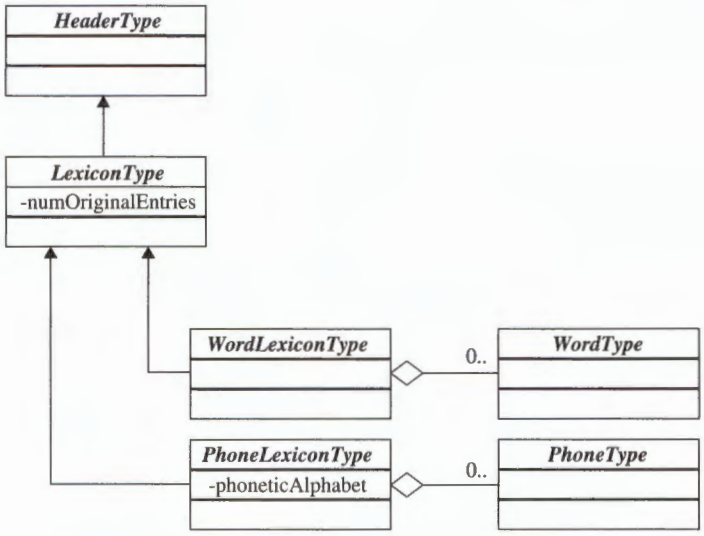


Figure 18.7 The word and phone lexicons are derived from the common LexiconType

those words that appear in the lattice and indicate the size of the original lexicon through an attribute value. This is illustrated in Figure 18.7.

The issue of what comprises a word in the lexicon is complicated because of the need to support a wide range of languages. Most western languages use the roman character set, but often augmented with accented characters that fall outside the normal ASCII character range. Eastern languages can be made up of entirely nonroman characters. Extensible Markup Language (XML) is capable of representing words using unicode, and we place no other restriction on the format of a word, other than that it cannot contain normal white space characters.

18.6.3 Phone Lexicon

The purpose of the phone lexicon is to represent the phone set used by one or more speakers. For any particular language, a number of phone sets are possible, and indeed, with the emergence of ASR systems, a plethora of different phone sets have been created through modification so as to enhance the accuracy of the ASR system. To ensure interoperability, we must attempt to restrict these phone sets to systems that are both international and parts of existing standards, without restricting the functionality of the underlying ASR.

Within the phone lexicon description scheme, two basic phone sets are supported via a phoneticAlphabetType: IPA (International Phonetic Association alphabet) and SAMPA (Speech Assessment Methods Phonetic Alphabet). The IPA [10] represents the basic sounds from which any of the world's languages may be articulated. However, the IPA alphabet, although widely used by phoneticians, can be cumbersome because of its reliance on unicode or numeric sequences. In such cases, SAMPA phone sets may be used. SAMPA [11] is an ASCII representation of IPA symbols, and was originally designed for a small number of European languages. More recently, it has been extended by several

distinct groups to cater for more specific needs. In the context of MPEG-7, we use 'SAMPA' to refer to whichever SAMPA inspired extension is appropriate; it is ultimately defined by the lexicon in the description. Unfortunately, silence is not considered a phone, and within ASR no standard exists for its representation. To account for silences within a phone decoding we have chosen to employ the SAMPROSA (Speech Assessment Methods Prosodic Approach) suggestion of '...' to represent silence. For instances in which SAMPA and IPA are still inappropriate an additional enumeration is 'unknown'; this is to indicate a nonstandard phone set and hence limits interoperability. It does, however, enable applications in which the actual phones used are irrelevant or are known to be the same as in the query engine.

18.6.4 Phone Confusion Statistics

The limited number of phones permits the inclusion of phone confusion statistics. These allow the derivation of conditional decoding probabilities, that is, $P(d|p)$, where p is the phone actually articulated and is the phone that was decoded. The statistics representative of the probability of the decoder inserting or omitting phones are also included. These statistics permit context independent statistical matching of a query with the annotations [12].

18.6.5 Speaker Information

The primary element of the Header is the Speaker Information, which links together the lexica and statistics. The Speaker Information element is designed to represent a single person who is speaking, although the actual person can be implied. When implied, the data structure serves as a root for associated references. The SpokenContentDescriptionScheme is designed to support multiple speakers with, potentially, different vocabularies, statistics or even languages. The following attributes and elements thus characterize a speaker:

1. *Person identification*: This takes the form of the person's name(s) and title(s). Ideally, this should be unique across all the speakers in the header.
2. *References to the word and/or phone lexica*: The transcription representing a single speaker may be represented in words as well as phonetically. A speaker may use a maximum of one word lexicon and one phone lexicon.
3. Reference to the confusion statistics representing the phone confusability of this speaker's phone decodings. It is worth noting that although two speakers may use the same phone set, user adaptation means their confusion statistics may differ.
4. *The provenance*: This is an enumerated type listing the source of the speaker's annotation. Permitted values are 'unknown', 'ASR', 'manual', 'keyword' and 'parsing'.
5. *The spoken language*: This represents the language and dialect spoken using the ISO-3166 and ISO-639 standards.
6. Description metadata, among other things, allows the extraction tool used and its settings to be described. This is primarily intended for use within a corpus or database, as unstructured information such as this cannot be readily used to assist interoperability.

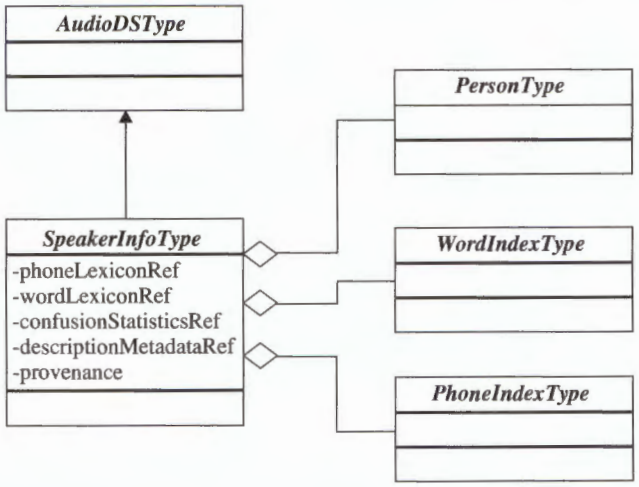


Figure 18.8 The speaker information forms the key element linking the words, phones and indexes together

7. Optional word and phone indexes using word and phone n-gram keys are included. Although, in principle, redundant information, their inclusion permits hierarchical retrieval over networks.

The structure of the SpokenContent Speaker Information is illustrated diagrammatically in Figure 18.8.

18.6.6 SpokenContent Lattice

While the SpokenContent Header contains explanatory metadata, the content of an annotation resides in the SpokenContent Lattice, the structure of which is shown in Figure 18.9.

The purpose of the lattice is to represent the spoken content in a manner capable of supporting the output of the majority ASR systems, while also containing sufficient information that robust retrieval is possible. As discussed above, the ideal approach is to represent the output of the extraction tool in the form of a combined word and phone lattice (see, Figure 18.3). The lattice consists of links, nodes and blocks.

Word and phone links

The smallest component of the lattice is the link. A link joins two nodes and represents a single word or phone. It contains a probability and node offset (indicating the target node – the node at which the link terminates).

Lattice nodes

A node represents a point in time at which a speech event (start or end of a word or phone) occurs. It contains the time and one or more word or phone links (which begin

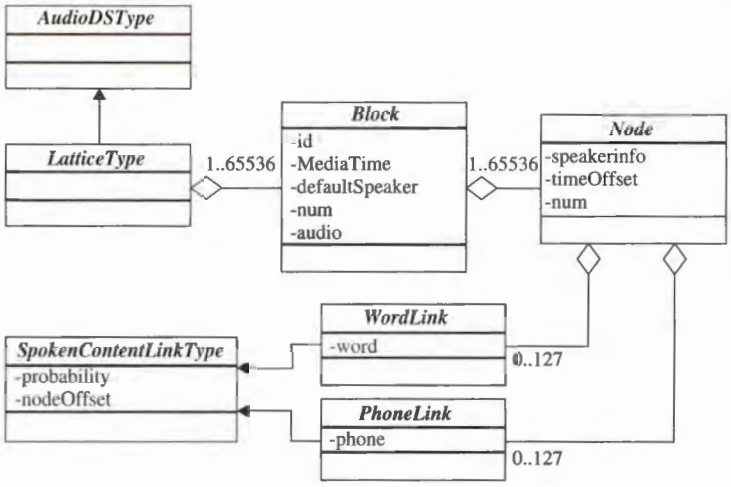


Figure 18.9 The lattice represents a single transcription or transcription segment. It represents a combined word and phone statistical lattice supporting speech from one or more speakers

at that node) and an optional speaker reference. A node is associated with only a single speaker. Where a lattice is predominantly comprises nodes from a single speaker, a default speaker may be set (see later) and the speaker attribute on the node omitted to improve memory efficiency.

It may be noted that the total probability of all word links or all the phone links emanating from a node must sum to one or less. A sum of less than one reflects the omission of some paths. However, the word and phone links are independent projections of the same probability space and their probabilities not trivially combinable.

Lattice blocks

The SpokenContent Lattice may, conceivably, represent many hours of spoken annotation. To facilitate interrogation of the SpokenContent lattice on platforms of limited memory, the lattice is subdivided into blocks. This permits relative, rather than absolute, indexing of nodes and times with a consequent saving of memory. The lattice block also provides a convenient granularity at which to provide a coarse estimate of audio quality.

A lattice block contains:

1. *Start time of the block*: This is the time from which the node times are offset, and permits the node offset times to be held to 1/100ths of a second accuracy with little memory overhead. Such accuracy may be required for use in spoken corpora.
2. *Default speaker*: In many cases, a block of the lattice will predominantly represent a single speaker. The defaultSpeaker attribute permits the speaker identifier associated with a node to be omitted in such circumstances.
3. *Noise*: In keeping with the headings in the rest of the list we have named the list head as Noise conditions, Please clarify if this is o.k. *conditions:* The importance associated with the extracted lattice depends on the prevailing noise conditions. The

audio attribute is an enumerated type with permitted values: 'unknown', 'speech', 'noise', 'noisySpeech'

A SpokenContent Lattice thus consists of blocks, each of which comprises nodes and links.

18.7 USAGE OF THE SPOKENCONTENTDESCRIPTIONSCHEME IN AN MPEG-7 DESCRIPTION

18.7.1 Header/Body Structure

The split into 'header' and 'body' parts has helped define how the disparate components of the metadata are handled. MPEG-7 defines a base hierarchy of abstract types from which all descriptors and description schemes inherit. The main reason for this is to define placeholders at the top-level of MPEG-7, but without saying exactly what information the placeholder is to contain. The lattice type inherits from a base AudioDescriptionSchemeType description scheme, and can hence be accommodated within certain specific higher-level description schemes designed to contain audio information (notably the AudioSegment, see Section 18.7.2). The SpokenContentHeader and some of its contents inherit from a special HeaderType (see Figure 18.6). A placeholder for such a header exists in most description schemes, which allows these header types to be accommodated at almost any point in an MPEG-7 description. In general, a description containing more than one SpokenContentDescriptionScheme can accommodate the header(s) at some appropriately scoped node in the description. In this context, 'appropriately scoped' could mean top level, or it could mean the deepest node that still contains the descriptors to which the header applies (in a data-inheritance sense).

18.7.2 Segmentation

Central to the MPEG-7 model of a description of any media is the concept of a segment. A segment is a delimited section of a piece of media. In the case of visual metadata, the segment can be quite complicated; it can refer to a temporal or spatial segment that may or may not be moving. The audio situation is slightly simpler in that only temporal segmentation makes sense. A temporal segment can, however, contain a segment decomposition, which is a hierarchy of smaller segments that may or may not overlap in time, and may or may not contain finer decompositions.

The SpokenContentLattice is designed to fit inside an AudioSegment. To this end it inherits from an AudiodeScriptionSchemeType (see Figure 18.9), which is aggregated by the AudioSegment. Although the constituent parts of SpokenContentDescriptionScheme do make sense stand-alone, extra functionality is available via the segment, not least the ability to take part in segment relations. A single edge of a relationship graph can carry the information that a particular segment annotates another MPEG-7 entity. This is the paradigm that enables the photo annotation application in MPEG-7 (Figure 18.5). In general, each segment in a segment decomposition accommodates a single SpokenContentLattice.

Where a segmentation tool has been used to separate a section of audio into segments, perhaps corresponding to a particular speaker, it makes sense to represent this segmentation using a segment decomposition. Where the segmentation is performed by the ASR engine, however, another option is to use the node and block structure of the lattice. The identity of the speaker can be stored on each node of the lattice enabling a segmentation within the lattice itself. This may be useful where the speaker identification is performed by the ASR.

The latter segmentation also has an advantage when a description is streamed across some channel. Multiple speakers represented by the same lattice structure are naturally streamed in parallel. The same segmentation represented by the segment decomposition would transmit each speaker sequentially.

18.7.3 Referencing

For reasons of storage efficiency, the SpokenContentDescriptionScheme uses two kinds of non-IDREF referencing:

1. *Array indices*: An array index is a positive integer that refers to the *n*th element of an array. Where the array size is known, the size of the index can be restricted. In particular, if the array has 65 536 or fewer entries then the index need only be 16 bits. This is the mechanism by which links in the lattice reference their target nodes, and entries in the word and phone indexes (in the header) reference nodes and blocks. Although this approach is efficient it reduces human readability. In general, however, the lattice component of the SpokenContentDescriptionScheme, is automatically generated, so readability is not an issue.
2. *uriReferences*: An uriReference is like an IDREF, but can reference an ID in a different instance document. The parser cannot check the validity of the reference, as it does not in general have access to the referenced document. In the particular case of the SpokenContentDescriptionScheme, the lattice and header may exist in separate documents. To this end, the speaker references in the lattice and the lattice references in the indexes (in the header) are of type uriReference.

18.7.4 Binarization

The binarization of MPEG-7 is a vast topic in many ways distinct from the descriptors and description schemes that comprise the metadata. Although we will not examine the binarization issue in depth here, it is nonetheless worth considering how a system utilizing the SpokenContentDescriptionScheme would be implemented. In particular, the (potentially) large number of words, nodes and links means that careful consideration must go into their design.

Without making unjustified assumptions, an unconstrained counting type must be implemented to the maximum range supported by an integer on the platform. In many cases only a small number of items are realistically possible. In the SpokenContentDescriptionScheme, where possible, data types have been strongly constrained so that the smallest integer or character string possible may be used.

18.8 SPOKEN DOCUMENT RETRIEVAL (SDR)

18.8.1 Literature

SDR is by no means a new subject. There is a large amount of literature concerning retrieval from textual documents that applies directly to spoken documents [13, 14]). In more recent years, research has emerged that applies specifically to SDR [4, 5, 15]. Current leading edge research in the area of SDR is represented by the TREC SDR track run by NIST [2]; although the 1999 results are the most recent available in print at the time of writing, the competition has continued annually since then.

18.8.2 Example Evaluation

The validation of the data SpokenContentDescriptionScheme rests on the demonstration of two points. Firstly, that the structure is capable of storing output of an ASR engine in an efficient and usable manner and secondly that the increased complexity over, for example, a simple 1-best text improves retrieval rate. That is, it must be demonstrated that the combination of word and phoneme retrieval improves recall over either individually. To evaluate the structure of the SpokenContentDescriptionScheme described above, we used the simple document scoring:

$$S(d;q) = S_w(d;q) + S_s(d;q) + S_p(d;q)$$

Where d is a spoken document in MPEG-7 format, q is a textual query (converted to phones through a statistical text-to-phone system when necessary) and where

$$S_w(d;q) = \sum_{w_d, w_q} P(w_d)\delta(w_q, w_d)$$

w_d and w_q are words in the document and query respectively. The morphological term is given by

$$S_s(d;q) = \lambda \sum_{m_q = M(w_q), m_d = M(w_d)} P(w_d)\delta(m_q, m_d)$$

where $M(x)$ is a word stem after stop-word removal and morphological word stemming [16]. Lastly, the phone score is given by

$$S_p(d;q) = \eta \sum_{\vec{T}_q, \vec{T}_d} P(\vec{T}_d)O(\vec{T}_q, \vec{T}_d)$$

$\vec{T}_x$ is a phone trigram and $O(x, y)$ is a phone trigram overlap function. In our experiments we used a δ-function, although statistical overlap functions can be used, see, for example, [12]. The mixing parameters were set to $\lambda = \eta = 0.3$. This retrieval algorithm is deliberately not state of the art.

Table 18.1 Retrieval results for short annotation style spoken documents

Number of queries	Number of words in query					
	2		3		4	
	25		13		8	
	Recall	Precision	Recall	Precision	Recall	Precision
Word query	0.41	0.30	0.47	0.23	0.48	0.24
Phonetic query	0.43	0.13	0.52	0.14	0.62	0.18
Word + Phone query	0.51	0.17	–	–	–	–

Where the regime is known in advance, for example, large annotations, we may utilize such methods as query expansion to significantly enhance retrieval. Indeed, as the MPEG-7 SpokenContentDescriptionScheme retains all the information essential to state-of-the-art retrieval systems used in the TREC evaluations, we may expect their results to apply to this data structure. The purpose of our retrieval experiment is to demonstrate the importance of the combined word and phone lattice structure to retrieval even under difficult circumstances. This may be illustrated with a collection of short annotations one would typically find in audio captions. We employed a database of 2907 documents with an average duration of about 8.5 seconds. We used an ASR system with a 20 000-word vocabulary working to a word accuracy of 40% on the test database. The retrieval results are given in Table 18.1.

It is evident that the information contained in both the word and phone components of the lattice is essential and complementary. Where words prove to be OOV or misrecognized because of extraction errors or noise, phonetic matching can significantly boost retrieval performance. Where large annotations and queries are available and very high accuracy extraction systems are used, the contents of the spoken content lattice may be pruned back to just the best few hypotheses. Consequently, the SpokenContentDescriptionScheme component of MPEG-7 can support a wide range of audio quality and content by the added value of the probabilities and multilevel representation.

18.9 SUMMARY

We have demonstrated that despite the inherent errors in ASR output, a workable spoken content description scheme may be constructed. By retaining the content in a multilevel representation using both words and phones, we can support retrieval even where the extraction tool was in error. Through the retention of statistical information, both in the lattice and in the confusion statistics, characterizing the errors in the extraction tool, we may ensure interoperability across heterogeneous databases.

REFERENCES

[1] S. E. Johnson *et al.*, "Spoken document retrieval for TREC at Cambridge University," *Proceedings of TREC-6*, NIST special publication, 1998.

[2] E. Voorhees and D. K. Harman, eds. *"The Eighth Text REtrieval Conference (TREC-8),"* NIST Special Publication 500–246, Department of Commerce, National Institute of Standards and Technology, Gaithersburg, Md., 1999.

[3] J. P. A. Charlesworth and P. N. Garner, "Spoken content metadata and MPEG-7," *Proceedings of ACM MM2000 workshops*, 81–84 (2000).

[4] K. Ng and V. W. Zue, "Subword unit representations for spoken document retrieval," *Proceedings of Eurospeech*, (3), 1607–1610 (1997).

[5] K. Ng and V. W. Zue, "Phonetic recognition for spoken document retrieval," *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP '98, vol. 1, Seattle, WA, USA, 1998, pp. 325–328.

[6] K. Ng and V. W. Zue, "Subword-based approaches for spoken document retrieval," *Speech Communication*, **32**, 157–186 (2000).

[7] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, John Wiley & Sons, New York, 1993.

[8] I. Bazzi and J. R. Glass, "Modeling out of vocabulary words for robust speech recognition," *Proceedings of ICSLP*, 401–404 (2000).

[9] A. T. Lindsay *et al.*, "Representation and linking mechanisms for audio in MPEG-7," *Image Communication*, **16**, 193–209 (2000).

[10] International Phonetic Association, *"Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet,"* ISBN 0–521–65236–7, Cambridge University Press, New York, UK, 1999.

[11] D. Gibbon, R. Moore and R. Winski, eds. *"Handbook of Standards and Resources for Spoken Language Systems,"* ISBN 3–11–015366–1, Mouton de Gruyter, Berlin, New York, 1997.

[12] S. Srinivasan and D. Petkovic, "Phonetic confusion matrix-based spoken document retrieval," *Proceedings of SIGIR*, 81–87 (2000).

[13] I. H. Witten, A. Moffat and T. C. Bell, *"Managing Gigabytes,"* Van Nostrand Reinhold, New York, 1994.

[14] D. A. James, The Application of Classical Information Retrieval to Spoken Documents, PhD thesis, Cambridge University, New York, 1995.

[15] J. Foote, "An overview of audio information retrieval," Multimedia Systems 3, Springer, Berlin, 1999, pp. 2–10.

[16] M. F. Porter, "An algorithm for suffix stripping," *Program*, **14**(3), 130–137 (1980).

19

Sound Classification and Similarity

Michael A. Casey

Mitsubishi Electric Research Labs, Cambridge, USA

General audio consists of a wide range of sound phenomena such as music, sound effects, environmental sounds, speech and nonspeech utterances. The sound recognition tools provide a means for classifying and querying such diverse audio content using probabilistic models. This chapter gives an overview of the tools and discusses applications to automatic content classification and content-based searching.

The sound classification and indexing tools are organized into low-level descriptors (LLD), *AudioSpectrumBasis* and *AudioSpectrumProjection*, and high-level description schemes, *SoundModel* and *SoundClassificationModel*, which are based on the *ContinuousHiddenMarkovModel* and *ProbabilityClassificationModel* description schemes defined in the *Multimedia Description Schemes* (MDS) document. The tools provide for two broad types of sound description; text-based description by class labels and quantitative description using probabilistic models. Class labels are called *terms* and they provide qualitative information about sound content. Terms are organized into classification schemes, or *taxonomies*, such as *music genres* or *sound effects*. Descriptions in this form are suitable for text-based query applications, such as Internet search engines, or any processing tool that uses text fields. In contrast, the quantitative descriptors consist of compact mathematical information about an audio segment and may be used for numerical evaluation of sound similarity. These latter descriptors are used for audio query-by-example (QBE) applications. They can be applied to many different sound types because of the generality of the low-level features. We start by discussing these LLD.

19.1 SPECTRAL BASIS FUNCTIONS

The frequency spectrum is a widely used tool in audio signal processing applications. However, the direct spectrum is generally incompatible with automatic classification methods because of its high dimensionality and significant variance for perceptually similar signals. Consider, for example, the *AudioSpectrumEnvelope* descriptor in which each spectral frame is an n-dimensional vector. A one-fourth – octave spectrum requires around 32 dimensions per frame at a sample rate of 22.05 kHz, but probability classifiers require data to occupy 10 dimensions or fewer for optimal performance. A lower resolution

representation, such as the octave bandwidth spectrum, reduces the dimensionality to 8 coefficients but disregards the narrow-band spectral information essential to many sound types. The solution is a representation that performs a trade-off between dimensionality reduction and information loss. This is the purpose of the *AudioSpectrumBasis* and *AudioSpectrumProjection* low-level audio descriptors.

The *AudioSpectrumBasis* descriptor contains basis functions that are used to project spectrum descriptions into a low-dimensional representation. The basis functions are decorrelated features of a spectrum with the important information described much more efficiently than by the direct spectrum. The reduced representation is well suited for use with probability models because basis projection typically yields useful features that consist of 10 dimensions or fewer.

In the simplest extraction method, basis functions are estimated using the *singular value decomposition* (SVD). The SVD is a well-known technique for reducing the dimensionality of data while retaining maximum information content. The SVD decomposes a spectrogram, or collection of spectrograms belonging to a sound class, into a sum of vector outer products with vectors representing both the basis functions (eigenvectors) and the projected features (eigen coefficients). These basis functions and projection coefficients can be combined to form eigenspectra, which are, in themselves, complete spectrograms but they contain mutually exclusive subsets of the original information. The full complement of eigenspectra sum to the original spectrogram with no information loss.

A subset of the complete basis is selected to reduce spectrum dimensionality. The loss of information is minimized because the basis functions are ordered by statistical salience; thus, functions with low information content are discarded. Figure 19.1 shows

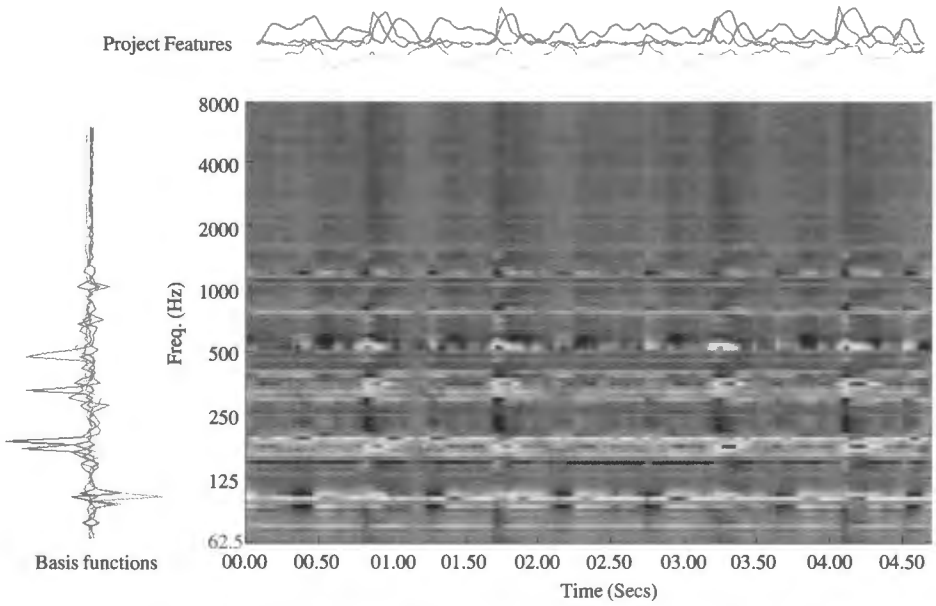


Figure 19.1 A four basis component reconstruction of a spectrogram of pop music. The left vectors are basis functions and the top vectors are the corresponding projected features

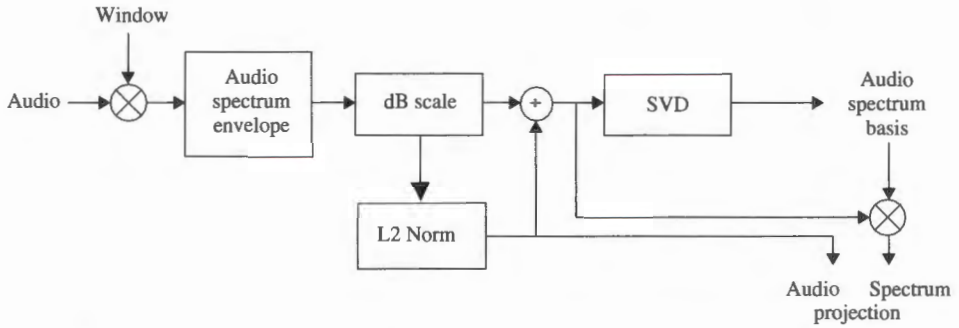


Figure 19.2 System flow diagram of basis function and projection coefficient extraction. The normalized spectrum is multiplied by a subset of the extracted basis functions to yield low-dimensional features

a spectrum of 5 seconds of pop music reconstructed using the first 4 out of 32 basis functions.

The functions to the left of the spectrogram are the basis functions, those above the spectrogram are the projected features that are used for automatic classification. Here, 70% of the original 32-dimensional data is captured by the 4 sets of basis functions and projection coefficients. Figure 19.2 shows the extraction system diagram for both *AudioSpectrumBasis* and *AudioSpectrumProjection*. Note that for classification applications, the normalization coefficients are stored with the projected features thus increasing the dimensionality by 1.

While the SVD is sufficient for most applications, an optional *independent component analysis* (ICA) step performs a transformation of the SVD basis functions yielding maximum separation of features. The ICA transform preserves the structure of the SVD feature space while offering a statistically independent *view* of the data; independence is a stronger condition than the de-correlation constraint imposed by the SVD [1–4].

The following code example shows an instance of the *AudioSpectrumBasis* descriptor for a 32-dimensional spectrum using a subset of 4 basis functions:

```
<AudioDescriptor xsi:type="AudioSpectrumBasisType"
loEdge="62.5" hiEdge="8000" octaveResolution="1/4">
<SeriesOfVector totalNumOfSamples="1" vectorSize="32 4">
<Raw dim="32 4">
0.082 -0.026 0.024 -0.093
0.291 0.073 0.025 -0.039
0.267 0.062 0.030 -0.026
0.267 0.062 0.030 -0.026
0.271 -0.008 0.039 0.007
0.271 -0.008 0.039 0.007
0.269 -0.159 0.062 0.074
<-- more values here ... -->
0.010 -0.021 0.063 -0.103
</Raw>
</SeriesOfVector>
</AudioDescriptor>
```

The next code example shows an instance of the *AudioSpectrumProjection* descriptor representing features derived from the basis vectors in the previous example:

```
<AudioDescriptor xsi:type="AudioSpectrumProjectionType">
<SeriesOfVector hopSize="PT10N1000F" totalNumOfSamples="263"
vectorSize="4">
<Raw dim="263 4">
0.359 -0.693 0.345 -0.145
0.364 -0.690 0.308 -0.147
0.353 -0.656 0.382 -0.175
<-- more values here ... -->
0.998 -0.342 0.569 0.592
1.000 -0.324 0.562 0.601
</Raw>
</SeriesOfVector>
</AudioDescriptor>
```

19.2 SOUND CLASSIFICATION MODELS

The first step toward automatic classification is to define a set of categories and their relationships. The *SoundClassificationModel* description scheme consists of sound models, each with an associated category term, organized into a hierarchical tree; for example, *people*, *musical instruments* and *animal* sounds. Each of these classes can be broken into narrower categories such as: *people:female*, *animals:dog* and *instruments:violin*.

Figure 19.3 shows musical instrument controlled terms that are organized into a taxonomy with 'Strings' and 'Brass'. Each term has at least one relation link to another term. By default, a contained term is considered a narrower term (NT) than the containing term. In this example, 'Fiddle' is defined as being a nearly synonymous with, but less preferable than, 'Violin'. To capture such structure, the following relations are available as part of the ControlledTerm description scheme:

- BT – *Broader Term*: The related term is more general in meaning than the containing term;
- NT – *Narrower Term*: The related term is more specific in meaning than the containing term;

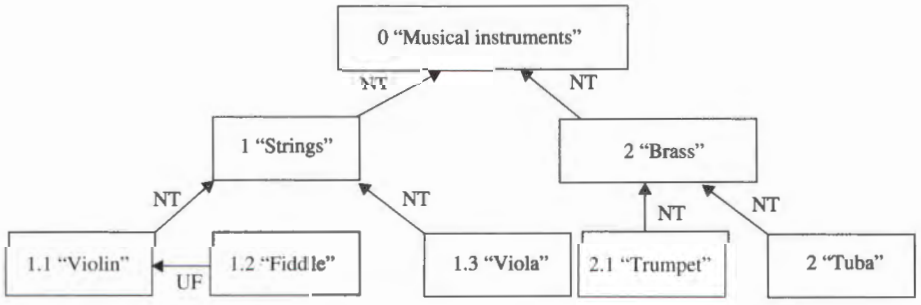


Figure 19.3 Part of a *Musical Instrument* classification scheme

- US – *Use*: The related term is (nearly) synonymous with the current term but the related term is preferred to the current term;
- UF – *Use For*: Use of the current term is preferred to the use of the (nearly) synonymous related term;
- RT – *Related Term*: Related term is not a synonym, quasi-synonym, broader or narrower term, but is associated with the containing term.

The purpose of the classification scheme is to provide semantic relationships between categories. As the scheme gets larger and more fully connected the utility of the category relationships increases. Figure 19.4 shows a larger classification scheme including *animal sounds*, *musical instruments, people* and *Foley* (sound effects for film and television). By descending the hierarchical tree we find that there are 19 leaf nodes in the taxonomy. By inference, a sound segment that is classified in one of the leaf nodes inherits the category label of its parent node in the taxonomy. For example, a sound classified as a *dog:bark* also inherits the label *animals*.

The following code example shows a number of *SoundClassificationModel* description scheme instances corresponding to the classification scheme shown in Figure 19.4. The classifiers are organized hierarchically with high-level classes preselecting the classifiers for low-level classes. Category terms are contained inside the *SoundModel* description scheme instances, see below. Here, references to *SoundModel* instances are used for brevity.

```
<!--General Audio Classifier, highest level categories-->
<AudioDescriptionScheme xsi:type="SoundClassificationModelType"
id="IDClassifier:GeneralAudio">
<SoundModel SoundModelRef="IDPeople"/>
<SoundModel SoundModelRef="IDMusicalInstruments"/>
<SoundModel SoundModelRef="IDAnimals"/>
<SoundModel SoundModelRef="IDFoley"/>
</AudioDescriptionScheme>
<!-- Human sound classes-->
<AudioDescriptionScheme xsi:type="SoundClassificationModelType"
id="IDClassifier:People">
<SoundModel SoundModelRef="IDSpeech:Male"/>
<SoundModel SoundModelRef="IDSpeech:Female"/>
<SoundModel SoundModelRef="IDCrowds:Applause"/>
<SoundModel SoundModelRef="IDPeople:FootSteps"/>
<SoundModel SoundModelRef="IDPeople:Laughter"/>
```

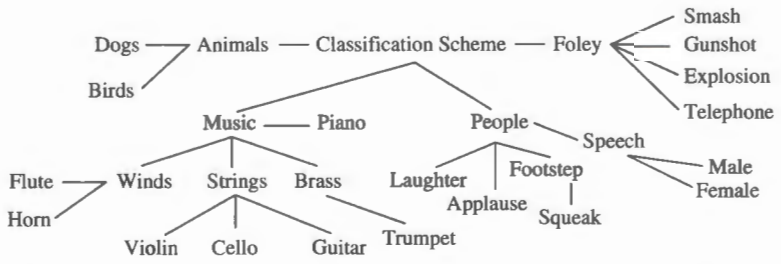


Figure 19.4 A hierarchical taxonomy consisting of *animals, music, people* and *Foley* classes

```
<SoundModel SoundModelRef="IDPeople:ShoeSqueaks"/>
</AudioDescriptionScheme>
<!-- Musical Instrument sound classes-->
<AudioDescriptionScheme xsi:type="SoundClassificationModelType"
id="IDClassifier:MusicalInstruments">
<SoundModel SoundModelRef="IDInstrument:Trumpet"/>
<SoundModel SoundModelRef="IDInstrument:Flute"/>
<SoundModel SoundModelRef="IDInstrument:Piano"/>
<SoundModel SoundModelRef="IDInstrument:Cello"/>
<SoundModel SoundModelRef="IDInstrument:Horn"/>
<SoundModel SoundModelRef="IDInstrument:Guitar"/>
<SoundModel SoundModelRef="IDInstrument:Violin"/>
</AudioDescriptionScheme>
<!-- Animal sound classes-->
<AudioDescriptionScheme xsi:type="SoundClassificationModelType"
id="IDClassifier:Animals">
<SoundModel SoundModelRef="IDAnimals:BirdCalls"/>
<SoundModel SoundModelRef="IDAnimals:DogBarks"/>
</AudioDescriptionScheme>
<!--Foley (film and television effects) sound classes -->
<AudioDescriptionScheme xsi:type="SoundClassificationModelType"
id="IDClassifier:Foley">
<SoundModel SoundModelRef="IDTelephones"/>
<SoundModel SoundModelRef="IDGunshots:Pistols"/>
<SoundModel SoundModelRef="IDExplosions"/>
<SoundModel SoundModelRef="IDGlass:Smashes"/>
</AudioDescriptionScheme>
```

19.3 SOUND PROBABILITY MODELS

Spectral features of a sound vary in time and it is this variation that gives a characteristic fingerprint for classification. Sound recognition models divide the sound feature space into a number of states and each state is defined by a continuous probability distribution. The states are labeled 1 to k, and sound is indexed by the most probable sequence of states for a given sound model. Figure 19.5 shows four states in two dimensions and a sound trajectory in the space. The dimensions correspond to the basis vectors contained in *AudioSpectrumBasis* and the trajectory corresponds to a sequence of spectral frames projected into an *AudioSpectrumProjection* descriptor.

The *SoundModel* description scheme is derived from the *ContinuousHiddenMarkovModel* description scheme defined in MDS. In addition to Markov model parameters, the *SoundModel* description scheme stores *AudioSpectrumBasis* functions that define the dimensions of the probability space, see the discussion of *AudioSpectrumBasis* above. The multidimensional Gaussian distribution is used for defining state densities. Gaussian distributions are parameterized by a $1 \times n$ vector of means, $\mathbf{m}$, and an $n \times n$ covariance matrix, $\mathbf{K}$, where n is the number of features (columns) in the observation vectors. The *GaussianDistribution* description scheme stores the *inverse* covariance matrix and the *determinant* of the covariance matrix along with the vector of means for each state. Therefore, the expression for calculating probabilities for a random vector, $\mathbf{x}$, given a mean vector, $\mathbf{m}$, covariance matrix inverse and the covariance matrix determinant is:

$$f_x(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}|\mathbf{K}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{m})^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{x}-\mathbf{m})\right]$$

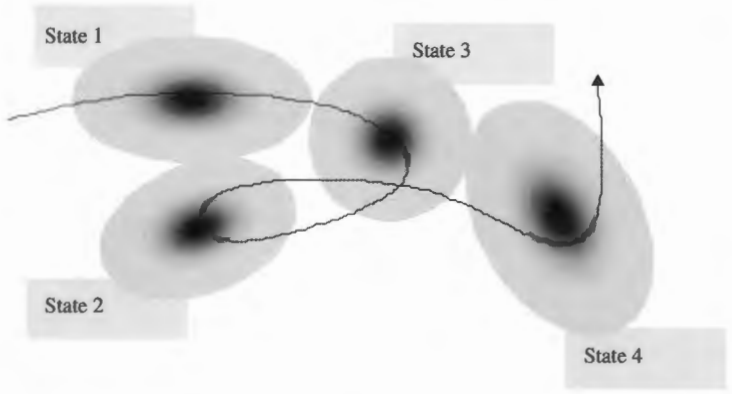


Figure 19.5 Four probability model states in a two-dimensional vector space. Darker regions have higher probabilities. A sound trajectory is shown by the line. The state parameters are chosen to maximize the probability of the states given a set of training data

The dynamic behavior of a sound model through the state space is described by a $k \times k$ transition matrix defining the probability of transition to each of the states from any current state, including the probability of self-transition. For a transition matrix, T, the ith row and jth column entry is the probability of a transition to state j at time t given state i at time $t - 1$. The transition matrix is a *stochastic matrix* that constrains the entries in each row to sum to 1. An initial state distribution, which is a $1 \times k$ vector of starting probabilities that also sum to 1, is required to complete the model. The kth element in the vector is the probability of being in state k in the first observation frame. The following example code shows an instance of a *SoundModel* for the *Trumpet* class that is contained in the *SoundClassificationModel* code example shown above. Floating-point numbers have been rounded to three decimal places for illustrative purposes.

```
<SoundModel id="IDInstrument:Trumpet">
<SoundClassLabel>
<Term id="ID16">Instrument:Trumpet</Term>
</SoundClassLabel>
<Initial dim="1 6"> 0.000 0.068 0.074 0.716 0.142 0.000 </Initial>
<Transitions dim="6 6">
1.000 0.000 0.000 0.000 0.000 0.000
0.000 0.994 0.000 0.000 0.000 0.006
0.000 0.000 0.993 0.007 0.000 0.000
0.014 0.000 0.095 0.818 0.000 0.074
0.000 0.000 0.000 0.005 0.995 0.000
0.056 0.000 0.000 0.000 0.000 0.944
</Transitions>
<DescriptorModel>
<Descriptor xsi:type="mpeg7:AudioSpectrumProjectionType"/>
<Field>SeriesOfVector</Field>
</DescriptorModel>
<State>
<Label>
<Term id="IDState1">State1</Term>
</Label>
<ObservationDistribution xsi:type="mpeg7:GaussianDistributionType">
```

```
<Mean dim="1 10"> 8.004 -4.805 4.850 5.738 1.261 -2.198 2.076 -0.324
-2.052 -0.022 </Mean>
<CovarianceInverse dim="10 10">
0.744 0.008 2.526 0.324 -0.049 -0.297 0.159 -0.074 -0.260 0.029
0.008 1.387 -1.087 1.906 0.838 -0.134 0.640 -0.321 -0.113 -0.053
2.526 -1.087 12.525 0.026 0.012 -0.002 0.009 -0.004 -0.002 -0.001
0.324 1.906 0.026 6.004 -0.020 0.003 -0.015 0.008 0.003 0.001
-0.049 0.838 0.012 -0.020 4.870 0.001 -0.007 0.003 0.001 0.001
-0.297 -0.134 -0.002 0.003 0.001 3.402 0.001 -0.001 -0.000 -0.000
0.159 0.640 0.009 -0.015 -0.007 0.001 3.157 0.003 0.001 0.000
-0.074 -0.321 -0.004 0.008 0.003 -0.001 0.003 1.816 -0.000 -0.000
-0.260 -0.113 -0.002 0.003 0.001 -0.000 0.001 -0.000 0.923 -0.000
0.029 -0.053 -0.001 0.001 0.001 -0.000 0.000 -0.000 -0.000 0.494
</CovarianceInverse>
<Determinant>0.339418</Determinant>
</ObservationDistribution>
</State>
<!-- Remaining States (2-6) similar to above . . . -->
<SpectrumBasis loEdge="62.5" hiEdge="8000" octaveResolution="1/4">
<SeriesOfVector totalNumOfSamples="1" vectorSize="31 9">
<Raw mpeg7:dim="31 9">
0.082 -0.026 0.024 -0.093 0.010 -0.021 0.063 -0.103 0.057
0.291 0.073 0.025 -0.039 0.026 -0.086 0.185 0.241 0.107
0.267 0.062 0.030 -0.026 0.054 -0.115 0.171 0.266 0.240
0.267 0.062 0.030 -0.026 0.054 -0.115 0.171 0.266 0.240
0.271 -0.008 0.039 0.007 0.119 -0.067 0.033 0.165 0.175
0.271 -0.008 0.039 0.007 0.119 -0.067 0.033 0.165 0.175
0.269 -0.159 0.062 0.074 0.182 0.071 -0.194 0.054 -0.009
0.246 -0.306 0.048 0.148 0.199 0.163 -0.324 -0.048 -0.065
0.216 -0.356 -0.037 0.137 0.059 0.215 -0.242 -0.035 -0.052
0.187 -0.359 -0.183 0.067 -0.343 0.223 0.023 -0.002 0.000
<!-- Remaining values here . . . -->
</Raw>
</SeriesOfVector>
</SpectrumBasis>
</SoundModel>
```

19.4 TRAINING A HIDDEN MARKOV MODEL (HMM)

Figure 19.6 shows the flow diagram for a HMM. The states are called *hidden* because the state sequence is not known but the data generated by the states are known. The observable data must be used to infer the position and extent of the states in the feature space, as well as the parameters for the transitions and initial state distribution. Once these parameters are known, a HMM can be used to convert a sequence of feature vectors into an optimal sequence of states.

Models are acquired by statistical analysis of training data – this process is called *model inference*. The first, and most important, step in training a HMM is collecting a representative set of exemplars for a given sound class; for instance, *violins*. For the best results, a number of violin example sequences should be collected that are recorded in differing environments and at many pitches. Ideally, the training set should contain all of the variation for the sound class. A set of 100 sounds is a good guide for training a sound class. However, some sound types exhibit more variation, such as *speech*, and therefore need more training examples to create a successful model.

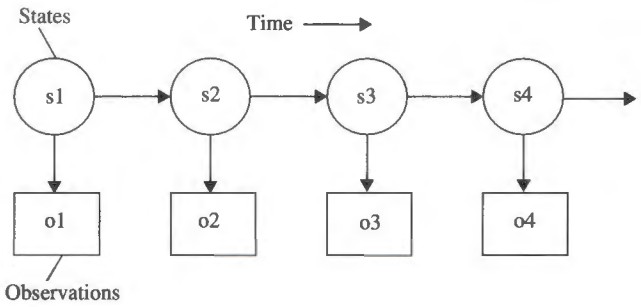


Figure 19.6 HMM system showing a sequence of states generating a sequence of observations. HMM inference performs the inverse mapping from training data to state parameters

During training, all of the parameters for a HMM must be estimated from the feature vectors of the training set. Specifically, these parameters are the numbers of states, the mean vector and covariance matrix for each of the states, the initial state distribution and the state transition matrix. Conventionally, these parameters are obtained using the well-known *Baum – Welch* algorithm. The procedure starts with random initial values for all of the parameters and optimizes the parameters by iterative reestimation. Each iteration runs over the entire set of training data in a process that is repeated until the model converges to satisfactory values. For a detailed introduction to designing and training HMMs [5].

19.5 INDEXING AND SIMILARITY USING MODEL STATES

Indexing a sound consists of selecting the best-fit HMM in a classifier and generating the optimal state sequence, or *path*, for that model. The state path is an important method of description since it describes the evolution of a sound through time using a very compact representation; specifically, a sequence of integer state indices. Figure 19.7 shows the state path for a dog barking; there are clearly delimited onset, sustain and termination or silent states. In general, states can be inspected via the state-path representation and a useful semantic interpretation can often be inferred for a given sound class.

Dynamic time warping (DTW) and histogram sum-of-squared differences are two methods for computing the similarity between state paths generated by a HMM. DTW uses linear programming to give a distance between two functions in terms of the cost of warping one onto the other. We may apply DTW to the state paths of two sounds in order to estimate the similarity of their temporal structure [5].

However, there are many cases in which the temporal evolution is not as important as the relative balance of occupied states between sounds. This is true, for example, with sound textures such as rain, crowd noise and music recordings. For these cases it is preferable to use a temporally agnostic similarity metric such as the sum-of-squared differences between state-path histograms. The *SoundModelStatePath* descriptor provides a container for these histograms. Frequencies are normalized counts in the range 0 to 1

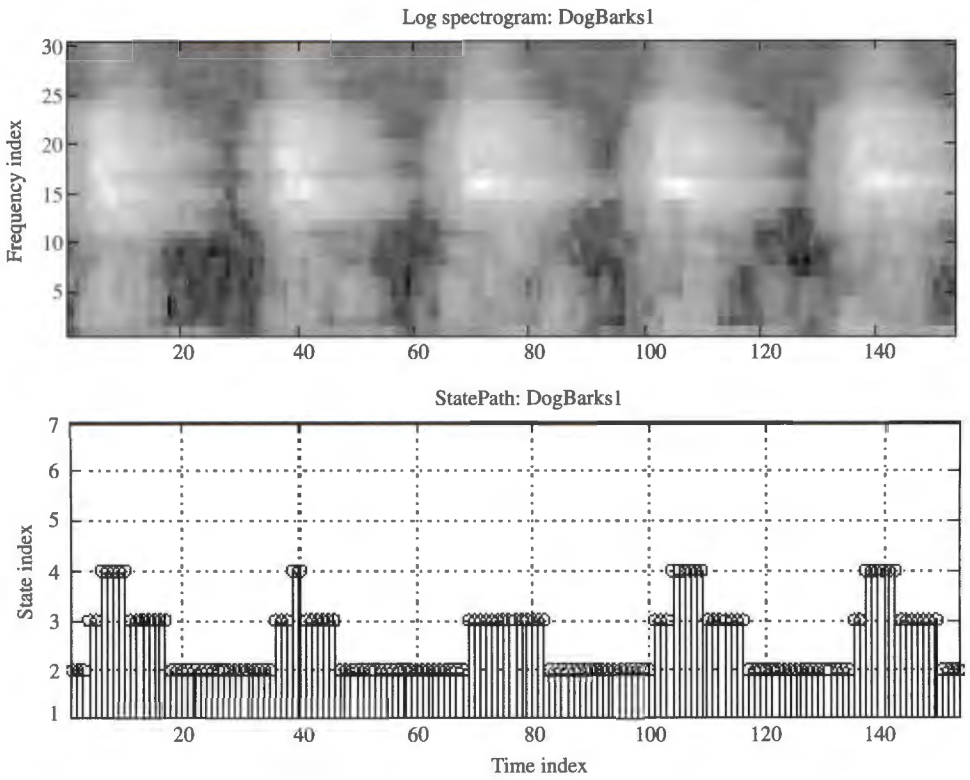


Figure 19.7 *AudioSpectrumEnvelope* of a dog barking and *SoundModelStatePath* generated by a dog bark HMM.

obtained by dividing the counts for each state by the total number of samples in the state sequence:

$$\text{hist}_a(j) = \frac{N(j)}{\sum_{i=1}^{K} N(i)}, \quad 1 \leq j \leq K$$

where K is the number of states, $N(j)$ is the count (frequency) for state j for the given audio segment. Similarity is computed as the absolute difference in the relative frequency of each state between different sound instances. The differences are summed to give an overall distance metric. For two sound models a and b, the distance is defined as:

$$\delta(a, b) = \sum_{j=1}^{k} \sqrt{[\text{hist}_a(j) - \text{hist}_b(j)]^2}$$

This similarity method applies for all classes of sound and therefore constitutes a generalized sound similarity framework [6, 7]. The following section gives examples of these methods applied to automatic sound classification and QBE searches.

19.6 SOUND MODEL APPLICATIONS

19.6.1 Automatic Audio Classification

Automatic audio classification finds the best-match class for an input sound by presenting it to a number of HMMs and selecting the model with the highest *likelihood* score. A combination of HMMs used in this way is called a *classifier*. To build a classifier, a set of individual *SoundModel* description schemes are trained, one for each class in a classification scheme, and combined into a *SoundClassificationModel*. Given a query sound, the spectrum envelope is extracted and the result is presented to each sound model. The spectrum is projected against the model's basis functions producing a low-dimensional feature representation represented by the *AudioSpectrumProjection* descriptor. The *Viterbi* algorithm is then used to compute the *SoundModelStatePath* and likelihood score and the HMM with the maximum likelihood score is selected as the representative class for the sound. The algorithm also generates the optimal state path for each model given the input sound. The state path corresponding to the maximum likelihood model is stored and used as an index for query applications. Figure 19.8 illustrates the method for automatic audio classification using a set of models organized into a *SoundClassificationModel* description scheme. For a detailed description of the Viterbi algorithm see [5].

In the following example, 19 HMMs were trained corresponding to the leaf nodes of the classification scheme shown in Figure 19.4 above. The database consisted of 2 500 sound segments divided into 19 training and testing sets. 70% of the sounds were used for training the HMMs and 30% were used to test the recognition performance. For details on the training techniques see [7–10].

The results of classification on the testing data are shown in Table 19.1. The results indicate good performance for a broad range of sound classes. Of note is the ability of the classifier to discriminate between speech sounds and nonspeech sounds, it can also distinguish between male and female speakers. This speech and nonspeech classifier can be used to increase the performance of automatic speech recognition (ASR) in the context of nonspeech sounds and to generate labels and indexes for all of the classes defined by the classifier.

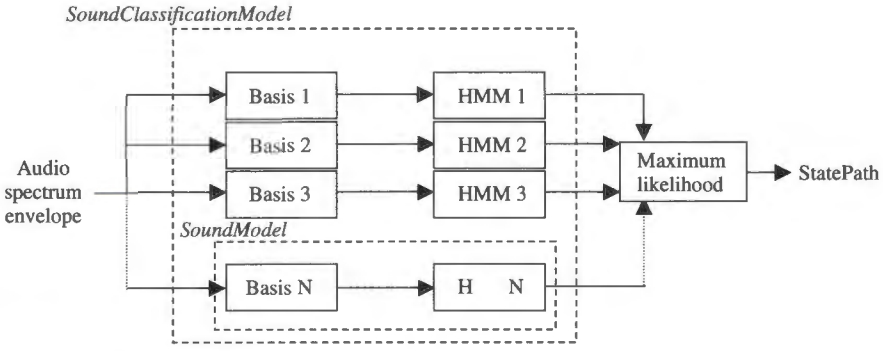


Figure 19.8 Maximum Likelihood Classification with the *SoundClassificationModel description scheme*

Table 19.1 Performance of 19 classifiers trained on 70% and cross-validated on 30% of a database consisting of 2 500 sound clips between 1 second and 30 second duration

Model name	% Correct classification
[1] AltoFlute	100.00
[2] Birds	80.00
[3] Piano	100.00
[4] Cellos (Pizz and Bowed)	100.00
[5] Applause	83.30
[6] Dog Barks	100.00
[7] Horn	100.00
[8] Explosions	100.00
[9] Footsteps	90.90
[10] Glass Smashes	92.30
[11] Guitars	100.00
[12] Gun shots	92.30
[13] Shoes (squeaks)	100.00
[14] Laughter	94.40
[15] Telephones	66.70
[16] Trumpets	80.00
[17] Violins	83.30
[18] Male Speech	100.00
[19] Female Speech	97.00
Mean recognition rate	**92.646**

The second classification example describes an experiment in music genre classification using the feature extraction and training methodology outlined above. Several hours of material were collected from compact discs and MPEG-1 (Layer III) compressed audio files corresponding to eight different musical genres. The data was split into 70/30% training/testing sets and HMMs were trained in the same manner as described above. Each sound file was split into several chunks consisting of a maximum of 30-second segments. Thus, the models were tuned to capture localized structures in the sound data. Table 19.2 shows the results of classification into music genres for the novel testing data. These results indicate that the classification system generalized to recordings consisting of mixed sources.

19.6.2 Audio QBE

In a QBE system, feature extraction is performed on a query sound as shown in Figure 19.8 above. The model class and state path is stored and these results are compared against the state paths stored in a precomputed sound index database using the sum of square differences in relative state frequencies also discussed above. Figure 19.9 shows a screen shot of an application called *SoundSpotter* that uses the sound classification tools to search a large database by categories and finds the best matches to the selected query sound using state-path histograms. Figure 19.10 shows the query and best-match state-path histogram, as represented by the *SoundModelStateHistogram* description scheme. The frequency

Table 19.2 Performance of eight classifiers using a 70/30% training/testing split for music genre classification

Model name	% Correct classification
[1] Bluegrass	96.8
[2] Reggae	92.5
[3] Rap	100.0
[4] Folk	92.3
[5] Blues	98.7
[6] Country	88.9
[7] Gospel	95.7
[8] NewAge	98.3
Mean recognition rate	**95.4%**

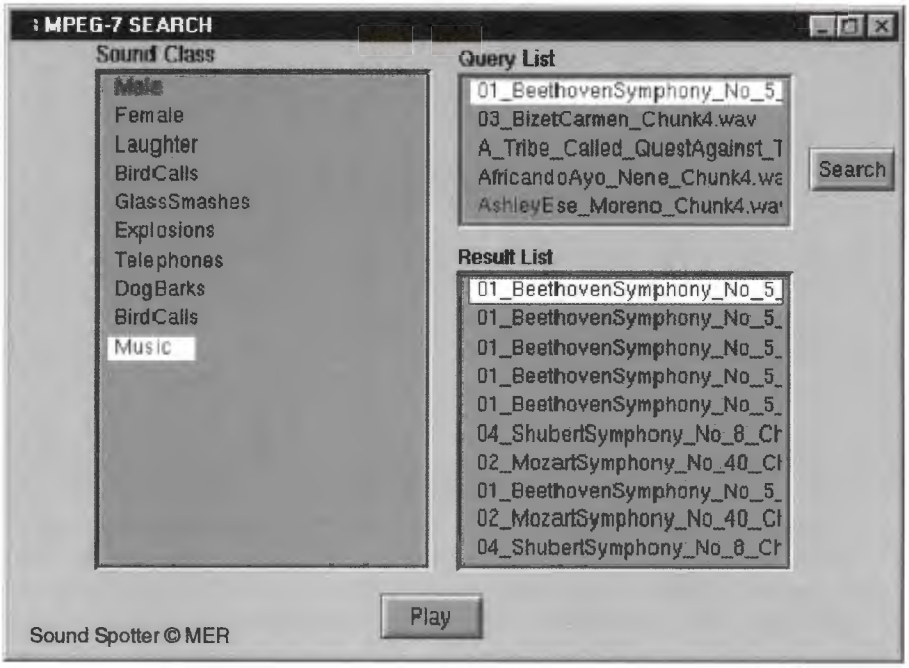


Figure 19.9 The *SoundSpotter* application finds the best matches to a highlighted query in a selected class. The 10 best matches are returned by the application on the basis of the distance between query and target state-path histograms. The categories on the left of the figure were automatically assigned by HMM classification

values are normalized to sum to 1 for each histogram, thus, sound segments of different lengths can be compared using these methods.

The examples given above organize similarity according to a taxonomy of categories. However, if a noncategorical interpretation of similarity is required a single HMM can be trained with many states using a wide variety of sounds. Similarity may then

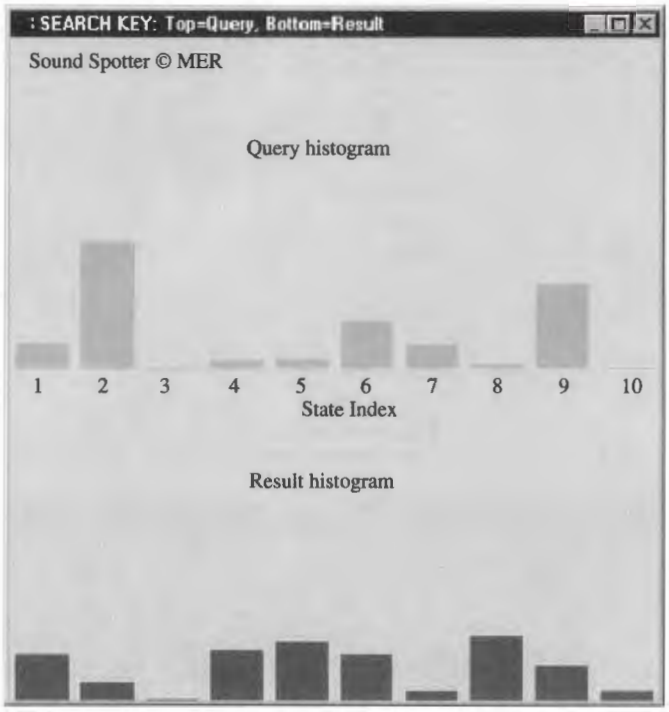


Figure 19.10 State-path histogram of a query sound and the best match found by the *SoundSpotter* application. Histograms are normalized so that the frequency values sum to 1

proceed without category constraints by comparing state-path histograms in the large generalized HMM state space.

19.7 SUMMARY

This chapter outlined the LLD and high-level description schemes for automatically classifying and querying sound content. The descriptors consist of low-dimensional representations of audio spectra that are extracted using linear basis methods. The high-level tools are based on classification schemes and HMM classifiers, both defined in MDS, and they are used to generate category labels for audio content and to index the content by HMM state indices.

One of the major design criteria for the tools was to represent a wide range of acoustic sources including textures and mixtures of sound. As such, the tools presented herein exhibit good performance on musical sound recordings as well as traditionally nonmusical sources such as vocal utterances, animal sounds, environmental sounds and sound effects. Use of the tools for automatic classification and QBE was discussed as well as extensions to generalized audio similarity.

In conclusion, the descriptors and description schemes outlined in this paper provide a consistent framework for analyzing, indexing and querying diverse sound types. The ability to automatically classify sounds, and the ability to search for *sounds-like*

candidates in a large database, independent of the source type, will be valuable components in new Internet music software, professional sound-design software, composers tools, audio-video search engines and many yet-to-be-discovered applications.

REFERENCES

[1] A. J. Bell and T. J. Sejnowski, An information-maximization approach to blind separation and blind deconvolution, *Neural Computation*, **7**, 1129–1159 (1995).

[2] J. F. Cardoso and B. H. Laheld, Equivariant adaptive source separation, *IEEE Transaction On Signal Processing*, **4**, 112–114 (1996).

[3] A. Hyvarinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transaction On Neural Networks*, **10**(3), 626–634 (1999).

[4] M. A. Casey and A. Westner, Separation of mixed audio sources by independent subspace analysis, *Proceedings of the International Computer Music Conference*, ICMA, Berlin, August 2000.

[5] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, N.J., 1993.

[6] M. Casey, "MPEG-7 sound recognition tools", *IEEE Transaction on Circuits and Systems Video Technology*, special issue on MPEG-7, IEEE, **11**(6), 737–747, 2001.

[7] M. Casey, Reduced-rank spectra and entropic priors as consistent and reliable cues for general sound recognition, Proceedings of the Workshop on Consistent & Reliable Acoustic Cues for Sound Analysis, EUROSPEECH 2001, Aalborg, September 2001.

[8] M. Brand, Structure discovery in conditional probability models via an entropic prior and parameter extinction, *Neural Computation*, **11**(5), 1155–1183 (1998).

[9] M. Brand, Pattern discovery via entropy minimization, Proceedings, Uncertainty '99, Society of Artificial intelligence and Statistics #7, Fla., 1999.

[10] J. Hershey and M. Casey, Audio-visual sound separation using hidden Markov models, *Advances in Neural Information Processing Systems*, Vol. 14, MIT Press, Cambridge, 2001.

Section VI
APPLICATIONS

20

Search and Browsing

Neil Day

MPEG-7 Alliance

20.1 INTRODUCTION

With the birth of the MPEG-7 standard, a new breed of multimedia applications has become possible. Even more exciting is that it places into the hands of increasingly more private citizens the ability to develop highly creative yet-to-be-seen applications. Most of us are familiar with searching for content on the Internet using text-based key word queries and till now this interface has more or less satisfied the needs of users. But as rich content becomes more widely available through digitization and being distributable over the Internet, the need for more efficient management and indexing of this content will certainly grow.

MPEG-7, based on Extensible Markup Language (XML) Schema, not only is used to index features of rich content at a catalogue (title) and semantic (who, what, where etc.) level but also at a signal level (timbre, pitch of audio, color and texture descriptors of still and moving images). By describing such features, the meaning and manipulation of the content becomes more accessible to users and will enable the generation of new unique applications. Hard-to-understand content features will be more meaningful and so their usage will be better understood.

Examples of content-based access include finding information using spoken queries, hand-drawn images and query by humming. For professionals, a new generation of applications will enable high-quality information search and retrieval such as television program producers wishing to retrieve occurrences of famous events or references to certain people, stored in thousands of hours of audiovisual records, in order to collect material for a program. The immediate benefits here would be reductions in program production time and increases in the quality of its content.

20.2 GROWTH OF RICH DIGITAL CONTENT

The MPEG-7 standard could not have come at a more timely juncture in the continuing growth of multimedia content available through the Internet. As more and more digital distribution systems are being produced with increasing bandwidth, the flow of rich content has also grown as users demand more audio and visual responses to their requests.

Additionally, in a major survey of the amount of information being produced in the world [1], the authors give estimates for the quantity of various types of content. According to this study, the world's total yearly production of print, film, optical and magnetic content would require approximately 1.5 billion GB (1.5 EB) of storage. This is the equivalent of 250 MB per person for each man, woman and child on earth.

In contrast, printed documents of all kinds comprise only 0.003% of the total. Magnetic storage is by far the largest medium for storing information and is rapidly becoming the universal medium for information storage. These numbers reinforce the need for improved content management systems and show clearly that search and retrieval of multimedia content will require more sophisticated solutions than is generally available today.

Increasingly, private citizens produce their own content and as the need for digitization of this content grows (e.g. photos and videos), households will have to manage terabytes of data. Indeed, of all information produced in the world, 93% is stored in digital form, whereas hard drives in stand-alone PCs account for 55% of the total storage shipped each year. In examining the amount of images being produced, the study found that more than 80 billion photographs are taken every year, which would take more than 400 petabytes to store, more than 80 million times the storage requirements for the text. (See *http://www.ccsf.caltech.edu/ roy/dataquan/* for a reference to memory size and comparisons).

20.3 MPEG-7 TO THE RESCUE

It is clear that not only is information growing enormously each year but the digitization of this information has become the preferred means of storage. With this growth of digital content and especially multimedia content, there is a critical need for more sophisticated content management applications that facilitate effective and efficient management, search and browsing of this content. MPEG-7 provides the tools needed to create these applications in an open and nonproprietary environment.

20.3.1 Some MPEG-7 Application Scenarios

Search and browsing performances will be more effective since the detail of content that can be described using MPEG-7 is quite comprehensive. Searching for content across databases, archives and the Internet will not only be done by simple keywords but also using queries made with typed and spoken keywords and phrases, still images, moving clips, sound effects and melodies. All content tagged with MPEG-7 metadata will be searchable and since the AV content can be tagged from the semantic level to the signal level, the user can expect rich and sophisticated responses to their queries. MPEG-7 will

make the web as searchable for multimedia content as it is searchable for text today. All domains making use of multimedia will benefit from MPEG-7, including

- digital libraries, education (image catalogue, musical dictionary, biomedical imaging catalogues...);
- multimedia editing (personalized electronic news service, media authoring);
- cultural services (history museums, art galleries etc.);
- multimedia directory services (e.g. yellow pages, tourist information, geographic information systems);
- broadcast media selection (radio channels, television channels);
- journalism (e.g. searching speeches of a certain politician using his name, his voice or his face);
- e-commerce (personalized advertising, on-line catalogues, directories of e-shops etc.);
- surveillance (traffic control, surface transportation, nondestructive testing in hostile environments etc.);
- investigation services (human characteristics recognition, forensics);
- home entertainment (systems for the management of personal multimedia collections, including manipulation of content, e.g. home video editing, searching a game, karaoke etc.);
- social (dating services).

Along with more rich textual queries (phrases and sentences), other types of queries made possible with MPEG-7 include the following:

- *Audio*: Search for songs by humming or whistling a tune or using an excerpt of Pavarotti's voice, get a list of Pavarotti's records and video clips in which Pavarotti sings or simply makes an appearance. Or play a few notes on a keyboard and retrieve a list of musical pieces similar to the required tune or images matching the notes in a certain way, for example, in terms of emotions.
- *Graphics*: Sketch a few lines on a screen and get a set of images containing similar graphics, logos and ideograms.
- *Image*: Define objects, including color patches or textures, and get examples from which you select items to compose your image. Or check if your company logo was advertised on a television channel as contracted.
- *Visual*: Allow mobile phone access to video clips of goals scored in a soccer game or automatically search and retrieve any unusual movements from surveillance videos.
- *Multimedia*: On a given set of multimedia objects, describe movements and relations between objects and so search for animations fulfilling the described temporal and spatial relations. Or describe actions and get a list of scenarios containing such actions.

The following sections outline a number of MPEG-7 applications that have been developed or are currently under development. They represent an insight to a new wave of applications that MPEG-7 will make possible where only creativity is the limiting factor. In some cases, the applications were being developed while the standard was being

developed, so they may not be completely compliant with MPEG-7 when it became an International Standard (IS) at the end of 2001.

The description of these applications is meant to give a conceptual overview and not to provide a deep technical explanation. The aim here is to provide an indication of typical applications that will be made possible by MPEG-7 tools.

20.4 REAL-TIME VIDEO IDENTIFICATION

This section presents a real-time video clip identification system using a high-speed video segment retrieval engine built using MPEG-7 visual technologies [2]. Similarity-matching functionality is one of the core technologies for a video archive management system in order to carry out signal-level video segment retrieval [3]. It helps users to find their preferred video program based on color, texture, motion and shape. Some typical uses of this real-time video identification include:

- investigation of broadcasted content for marketing activities;
- network monitoring of unauthorized content use;
- broadcast confirmation of the program;
- content filtering based on identical clips;
- immediate report of a broadcast problem.

Television commercial programs are one of the most promising target domains since not only their broadcast should be verified but also the investigation of the broadcast provides valuable feedback on a company's communications and marketing efforts. Real-time automatic monitoring systems are desirable by advertising agencies and marketing research companies. This is because, in general, they maintain their original monitoring systems relying heavily on manpower and involve high costs.

20.4.1 Using MPEG-7 Tools

The real-time video identification system uses the Color Layout Descriptor (CLD) [4] (see also Chapter 13). The CLD captures both the color distribution and the spatial layout of color. Figure 20.1 is an example of the video clip monitoring system using the CLD. The system captures live broadcasts and extracts the *Color Layout* data from each of the three frames. It forms a series of the *Color Layout* descriptions wrapped by a *Regular Visual Time Series* container, in which the time-increment element is set to three frames. The extracted description is compared with all of the series of the *Color Layout* descriptions stored in the database, and the ID number of the best-matched video clip is recorded in the log with broadcast time and location if its similarity is higher than the predefined threshold. The database also holds several associated textual data about the stored video clips; therefore, users can investigate the log file using semantic query information such as title, keywords, actors or actresses and summary. The database is automatically created from the live content with an additional operator's assistance to fill the semantic elements.

The system, built on an 800 MHz Intel Pentium III processor, can monitor more than 50 000 video clips in real time, whose lengths are typically 15 or 30 seconds. No

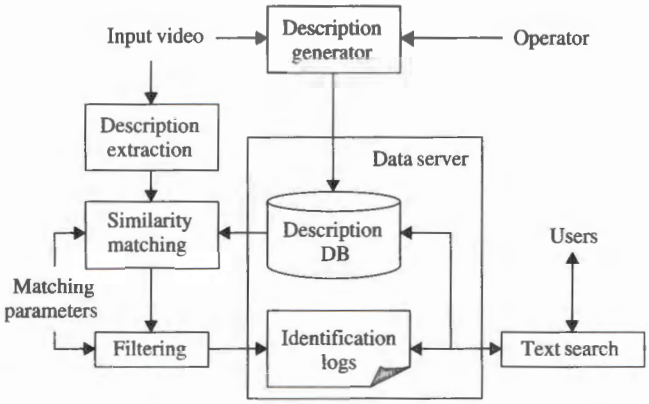


Figure 20.1 A block diagram of video clip monitoring system

overdetection and misdetection were found during the experiments that monitored over 56 hours of broadcasts in Tokyo. A test system has been continuously operated for more than 1 year (24 hours everyday). It monitors five commercial television broadcast channels to investigate all the commercial films broadcast in Tokyo, Japan.

20.5 QUERY-BY-HUMMING APPLICATION

Identifying musical works is a skill that comes fairly easy to most people. Often, when we turn on the radio in the middle of a familiar song, we are able to identify it within a few seconds (or even less). Development of this skill is taught in music classes (the so-called *drop-the-needle* test). This task, however, utilizes many different features of the music beyond the notes themselves such as lyrics, instrumentation, timbre, tempo and dynamics. What is more remarkable is that we can often identify music without those additional features and even without all of the notes, using melody alone. Even in many cases in which the melody is cut short, corrupted or inaccurately rendered, we can still identify what song is being referred to. A patron at a piano bar can convey a song to the pianist simply by humming the tune, often incorrectly. A skilled accompanist is able to take very imprecise queries and correctly infer the desired song. This task is known as *query by humming*, and components of MPEG-7 Audio (see Section V: Audio) are designed to facilitate this application.

Execution of this task highlights several features of melody. Melody is a very efficient construct. Just a few notes of a tune can fully identify a piece of music. It is also a fairly robust feature, resistant to corruption. Even when inaccurately recalled, a melody can often be identified. Identification by melody also scales easily to large data sets. Becoming familiar with more songs generally does not diminish our ability to identify ones we already know. Thus, it is desirable for any high-level representation of melody to also have these features (efficiency, robustness and scalability).

The MPEG-7 Audio standard includes two *Description Schemes* (representations) for melodic information. The first, *Melody*, is a complete and precise description of melody to facilitate search and retrieval using finely tuned queries. The Melody Description Scheme could be used by a trained musician to search for an exact sequence of

notes or intervals, perhaps entered by a keyboard. The second description scheme, called *MelodyContour*, was designed to facilitate the type of imprecise musical matching required by the query-by-humming application [5].

20.5.1 Using MPEG-7 Tools

Defining exactly what is or is not a melody can be somewhat arbitrary. Melodies can be monophonic, homophonic or contrapuntal [6]. Sometimes what one person perceives to be the melody is not what another perceives. A melody can be pitched or purely rhythmic, such as a percussion riff. In MPEG-7, the MelodyContour Description Scheme does not attempt to address all of these cases and is limited in scope to pitched, monophonic melodies.

Even with these limitations, arriving at a definition for melody can be difficult. Levitin [7] describes melody as 'an auditory object that maintains its identity under certain transformations... along the six dimensions of pitch, tempo, timbre, loudness, spatial location and reverberant environment; sometimes with changes in rhythm; but rarely with changes in contour'. Although broad, this definition highlights several important features of melody. People are able to recognize melodies even when they are played on different instruments, at different volumes and at different tempi (within a reasonable range) [8].

More importantly, for recognition purposes, a melody can still be uniquely identified after it has undergone transposition (we still recognize a familiar tune in a different key as being the same tune) [9]. For this reason, absolute pitch is not the best descriptor for melodic pitch information. What is important are the relative intervals between successive notes in a melody, since interval relations are also invariant to key transposition. In the query-by-humming application [10], however, we cannot assume that a query will contain precise and accurate intervals. A more robust feature is the *melody contour*, which is derived from interval information and is also invariant to transposition. The contour represents the general shape of the melody from note to note. Does the next note go higher or lower from the previous note or does it remain the same? Does it go much higher or just a little bit higher? On the basis of research and experimental evidence, a five-level contour (Table 20.1) was chosen for *MelodyContour* in MPEG-7, which is stored in a data type called *ContourData*, dividing the contour into small and large ascending and descending intervals with one level indicating no pitch change. For a sung or hummed query, this also reduces reliance on pitch tracking accuracy, which can be difficult to maintain over a wide variety of vocal ranges and timbres. Separation of the five contour levels is defined as follows:

Table 20.1 The five levels of contour information in MelodyContour Description Scheme

Contour value	Change in interval
−2	Descent of a minor-third or greater
−1	Descent of a half step or whole step
0	No change
1	Ascent of a half step or whole step
2	Ascent of a minor-third or greater

When identifying a melody, however, the listener perceives not only the interval and contour information in the melody but also how those notes correspond to particular moments in time (i.e. rhythm) [11, 12]. Rhythm is one dimension in which melodies in general cannot be transformed. The examples in Figures 20.2 and 20.3 illustrate the importance of rhythmic information in melodic description.
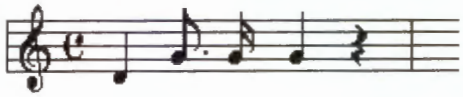


Figure 20.2 First four notes of the *Bridal Chorus* from *Lohengrin* (Wagner), that is *Here Comes the Bride*
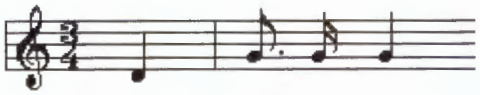


Figure 20.3 First four notes of *O Tannenbaum*

It is apparent that these are two very distinct melodies, yet the sounding pitches, intervals and note durations are identical. The difference lies not only in the respective meters (time signatures) of the songs but also in the position of the notes relative to the metric structure of each piece. The time signature of the first example is 4/4, and the strong beats occur on the first and third beats of a measure, which correspond to the first and fourth notes of the piece. The second example has a time signature of 3/4, and the strong beat is on the first beat of the measure, corresponding to the second note. From this example, we clearly see that a robust melody representation must also include rhythmic information. The *MelodyContour* Description Scheme stores both the meter of a melody and the nearest-whole-beat information for each note in the *MeterData* and *BeatData* data types, respectively. This rhythmic information disambiguates the above musical examples.

Figure 20.4 is another slightly more complex example from *Moon River* by Henry Mancini. To some people, the first interval is subject to uncertainty and is rendered incorrectly. Also, the sequence of descending whole and half steps can be easily confused. The *MelodyContour* Description Scheme has the facility to deal with such inaccuracies.



Figure 20.4 "Moon River"

20.5.2 Observations: Variations in Implementation

The MPEG-7 Audio standard specifies only the representations for data, but does not specify how the information (e.g. contour, meter and beat) is to be extracted or how

to compare such data. Automatic melody extraction from a mixed audio signal is an extremely difficult problem and is not covered by the standard. Likewise, comparison methods used in matching queries to a database of melodies are not standardized. Different distance metrics may be more accurate in differing situations, depending on the application. The goal of the *MelodyContour* Description Scheme is to provide an efficient, robust and easily scalable representation to enable search and retrieval comparisons in a framework flexible enough to accommodate different comparison technologies. This enables implementers to use proprietary technology in their competitive solutions for the marketplace.

20.6 CUIDADO

CUIDADO is an R&D project financed by the European Commission in the framework of the Information Society Technologies (IST) program that aims at developing content-based audio applications using MPEG-7 both for sound samples and music titles. There are four application domains targeted by CUIDADO technologies:

1. Database providers and Software engineering companies;
2. Sound and Music industry (radios, labels, music portals and music software companies developing databases and authoring tools);
3. Professional content creators (studios, archives and musicians using and processing a lot of sounds);
4. Electronic music distribution (anybody searching for music on the Web).

Two applications are being developed in CUIDADO, namely, a Music Browser and a Sound Palette [13, 14], using MPEG-7 Audio descriptions. The following sections outline the main functionalities of these applications and the type of MPEG-7 tools used to implement them.

20.6.1 Music Browser

The Music Browser includes sound-similarity search, constraint-based music selection, personal keyword usage and a user profiling [15]. One version is tied to web music monitoring for copyright societies and music labels and another to web music portals for record labels and web-customized radios, and both are combined with a user interface that enables the handling of a wide variety of listening behaviors. Functionalities include:

- *Sound similarity*: Input a given sound and get proposed matches (adjusting simple perceptual parameters such as 'attack' or 'pitch').
- *Music similarity*: Find music files that are sharing some high-level features such as music genre, melody or instrument played.
- *Search for music similarities*: With a captured audio file (from local files, web files, radios, mobile).
- *Constraint-based music selection*: Choose two music pieces and ask for a choice of music paths between both (or music compilations).

- *Multilingual keyword interface*: Link personal keywords (in any language) to a given sound or music. This learning system should then allow the user to input his or her personal keyword for retrieval purposes.
- *Automatic user's profiling*: By recording the metadata choices made in the previous sessions.
- *Fast listening*: *Based on salient features (Summary).*

20.6.2 Sound Palette

Expected to be used in digital music studios, the Sound Palette has an on-line version with retrieval and processing features scalable to network bandwidth performances and an off-line version enabling content-based editing and processing for existing professional authoring environments [16]. Typical functionalities include

1. in the new context of 'Giga samplers', the ability to make a fast sample retrieval by similarity, for replacing sounds or building new instruments, will be important;
2. in the orchestration context, benefits will be derived from the ability to make perceptual groupings according to pitch and timbres and to make sound edition based on content [17]: Classification and display according to pitch [18, 19]; Masking problem management; Use of symbolic tags and Timbre morphing;
3. ability to download files from an on-line database and to apply generic sound processing on a remote basis [20];
4. ability to map sound descriptors with image descriptors for multimedia applications.

20.6.3 Use of MPEG-7 Description Tools

MPEG-7 *Timbre* descriptors aim at describing perceptual features of instrument sounds [21, 22]. *Timbre* is currently defined in the literature as the perceptual features that make two sounds having the same pitch and loudness sound differently. The aim of the *Timbre* description tools is to describe these perceptual features using a reduced set of features such as 'attack', 'brightness' or 'richness' of a sound. The following low-level descriptors described by MPEG-7 metadata are provided to enable the (timbre) description of the perceptual features of sound: *Log-Attack Time, HarmonicSpectralCentroid, HarmonicSpectralDeviation, HarmonicSpectralSpread, HarmonicSpectralVariation, SpectralCentroid and TemporalCentroid* (see Chapter 17 for more details on these low-level audio descriptors).

The timbre descriptor for sustained harmonic sounds combines the above Spectral timbral low-level descriptors (LLD) with a log attack time descriptor. The percussive instrument descriptor combines the Timbral Temporal LLD with a spectral centroid descriptor. Comparisons between descriptions using either set of descriptors are done with an experimentally derived scaled distance metric.

Within four detailed possible classes of musical instrument sounds, as outlined in Table 20.2, two classes are well detailed. At this point, harmonic, coherent, sustained sounds (space 1) and nonsustained, percussive sounds (space 3) are represented

Table 20.2 Similarity spaces

Space number	1 Harmonic sounds	2	3 Percussive sounds	4
Sounds characteristics	1. Sustained 2. Harmonic 3. Coherent	4. Sustained 5. Nonharmonic 6. Coherent	7. NonSustained	8. Sustained 9. Noncoherent
Examples	Violin, flute, ...	Bell, triangle	Snare, calves	Cymbals, white

in MPEG-7. Additional descriptors from MPEG-7 include descriptors for sound effects, melody and silence.

20.7 TELEVISION NEWS PROGRAM APPLICATIONS

Two applications on news indexing and retrieval are introduced here. The first one is a news document model based on MPEG-7 developed by EPFL/EIF (Swiss Federal Institute of Technology/University of Applied Sciences Western Switzerland – EIF/Fribourg) and the second on filtering news articles developed by LG Electronics.

20.7.1 News Document Model Based on MPEG-7

This system enables users to manage different sources of information coming from news production, archiving and retrieval processes and uses an MPEG-7 implementation [23]. The application uses an XML-native database, TAMINO [24], for efficient storage, search and retrieval of news data.

This news document model contains descriptions (Table 20.3) of three phases, namely, *production*, *archiving* and *retrieval*. The main *production* descriptions are script (text read by the anchorperson), teletext subtitles, graphical overlay (text indicating the name of persons, places etc.) and finally the hierarchical structure of the news program. The *archiving* descriptions are mainly of two types: *visual description* and *contextual description*. *Visual description* is done either on shots or on a group of shots. It consists of free text describing the visual content including a set of keywords to indicate persons, companies, places and concepts seen in the video. *Contextual description* work is done on the news program and news items and describes in free text the event that has happened and also a set of keywords on persons, companies, places and concepts present in the event. A set of metadata on dates, anchorperson's and reporter's names and so on is also added. Also, during *retrieval*, users can add bookmarks and annotations to video segments.

MPEG-7 has been found to adequately represent all these various descriptions provided by different sources. The set of most important MPEG-7 tools used is shown in Table 20.3.

Table 20.3 Main MPEG-7 description schemes used to implement news model

TV news model	Corresponding in MPEG-7
Video segments (news program, news item, presentation, report etc.)	VideoSegment description scheme
News hierarchical structure	SegmentDecomposition on description scheme
Script	RelatedMaterial description scheme
Teletext subtitles, graphical overlay	VideoText description scheme
Free text description	TextAnnotation description scheme
Keywords descriptions	StructuredAnnotation description scheme

Architecture: A client-server architecture is used for this television news application. On the server side, all video content descriptions are stored in an MPEG-7 database (TAMINO). On the client side, the application provides two interfaces for indexing and retrieval. The indexing part is composed of different modules, each one specialized in producing a particular description of the video content. The retrieval part provides querying, browsing and visualization modules.

Indexing: This application supports the use of several indexing methods (automatic temporal segmentation, color histogram extraction, manual annotation etc.). For the semi-automatic indexing procedure, a shot detection and key frame extraction algorithm is applied to the video. The human indexer can then review the MPEG-7 results and add further manual annotations that augment the description of these results. The indexing interface is powered by the use of XSL (eXtensible Stylesheet Language, http://www/w3.org/Style/XLS) technology, which allows the user to create a desired personalized presentation of the news content.

Querying: The MPEG-7 descriptions are stored in an XML-native database, TAMINO [24]. This database provides a powerful query language, XML Query Language (XQL) [25], which allows Structured Query Language (SQL)-like queries on XML documents and enables a user to design an elaborate query model, which takes into account the structure, type and content of different descriptions.

Browsing: A browsing application that allows navigation inside a video news program based on the MPEG-7 TV news hierarchical structure has also been implemented. During browsing, other descriptions related to the video segments can also be visualized in order to get a better understanding of the video content.

In summary, this application provides an MPEG-7 model for television news content. Further information on this work can be found on the COALA project site (Content Oriented Audiovisual Library Access, *http://coala.epfl.ch*) [26, 27]

20.7.2 Filtering News Articles Using MPEG-7

This application provides an efficient way to filter articles that are of interest to the users, thus enabling the users to manage the content easily. It uses MPEG-7 description schemes associated with news broadcasts and enables users to selectively watch news articles that

are of interest to them. It is similar to reading text-based web newspapers where the readers skim through the whole news content by title or text summary and select only a small number of articles to read in detail. This TV News Browser can be used in a PVR (Personal Video Recorder) or in a PC.

An MPEG-7 description interface provides a function of 'selection' among an entire collection of content. In the case of news broadcasts, the structural information of news and other useful information that help the 'selection' process can be described by MPEG-7 description tools.

Application scenario

In general, a broadcast news program is composed of a number of articles. An article is an audiovisual explanation of an event that has happened in the world or of a homogeneous group of useful information data. Generally, an article is further composed of an anchor-shot and several episode-shots. An anchor-shot is the part of an article where the news anchor appears and explains briefly or introduces an event. Episode-shots are detailed explanations of an event.

The Article-based News Browser tool [28] uses the structural information of the news and additional information that helps users to browse news content. There are four representative key frames such as *anchor key frame*, *episode key frame*, *news icon* and *synthesized text*. The *anchor key frame* and *episode key frame* are representative frames for the anchor-shot and the episode-shot, respectively, while the *news icon* is a pictorial explanation of an article, and the *synthesized text key frame* is the synthesis of the videotext that is visible within the given article. These four representative key frames summarize an article and provide very useful information to help users understand an article at a glance.

Seeing the pictures, users can interpret roughly the stories in the article at a glance and playback the news articles of interest to them and go into the detail of the event.

Use of MPEG-7 description tools

The news content structure and the four representative frames are described in the Multimedia Description Schemes (MDSs) part of MPEG-7 standard [23]. The VideoSegment Description Scheme is used for the hierarchy of news content structure, which is constructed from an upper level VideoSegment containing the whole news program and a lower level VideoSegment containing each article. The SegmentDecomposition Description Scheme is used for specifying relations between the upper level and lower level VideoSegments. The four representative frames are described using the StillRegion Description Scheme and the SegmentRelationLink Description Scheme specifies the link between StillRegion data and VideoSegment data. Additionally, the MPEG-7 MediaLocator Description Scheme and TextAnnotation Description Scheme are used for describing pointers to actual video or images and the textual explanation of articles.

One shortcoming of this application is that the manual description of news content takes much time. However, as news content has a specific structure, it is possible to capture this structural information automatically unlike other content. Actually, automatic

indexing of news content, which extracts the start time and end time of articles and selects the representative frames, can perform considerably well. Providing textual annotation of the news articles can also be automated through speech-to-text and natural language processing technology.

20.8 MOVIE TOOL

This application is different from the ones discussed so far in that it is not, strictly speaking, an MPEG-7 application – that is, it does not enable search and navigation of AV content. The MovieTool is an MPEG-7 Description Authoring tool for Video with an easy-to-use visual interface. Although the tool is essentially used for indexing content, it should be noted that the ultimate purpose of such indexing is to enable improved search and navigation of content. MovieTool thus is an application that provides a rich set of tools for indexing content that can help search engine companies and content providers provide better-quality information services. The tool supports the composition of a logical structure of the input visual content and the edition of an output MPEG-7 file describing this content (see Figure 20.5). There are two steps involved, namely,

1. automatic or semiautomatic description using MPEG-7's *Segment Description Scheme* [23]
2. manual description using MPEG-7's *Usage /CreationInformation Description Scheme*, *TextAnnotation Datatype* [23] and so forth.

20.8.1 MPEG-7 Authoring Tool

The logical structure composer is used to create a logical structure hierarchy for content (see Figure 20.6). This composer provides some segmentation methods such as a cut detection algorithm.

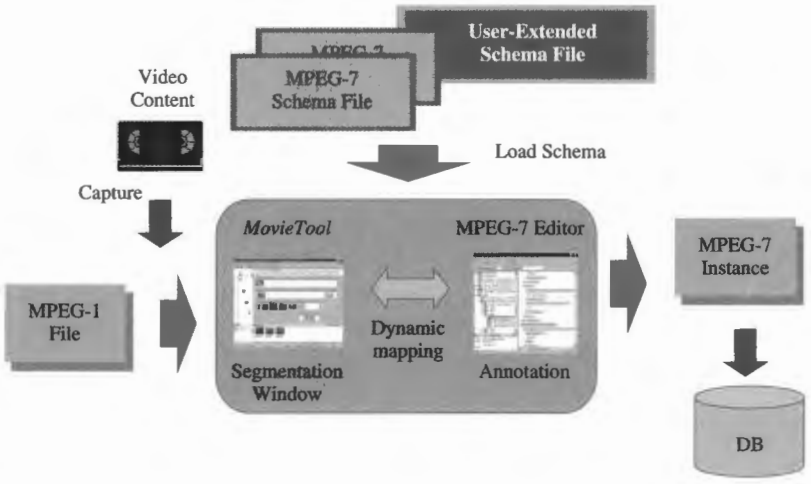


Figure 20.5 MovieTool: An MPEG-7 description authoring tool

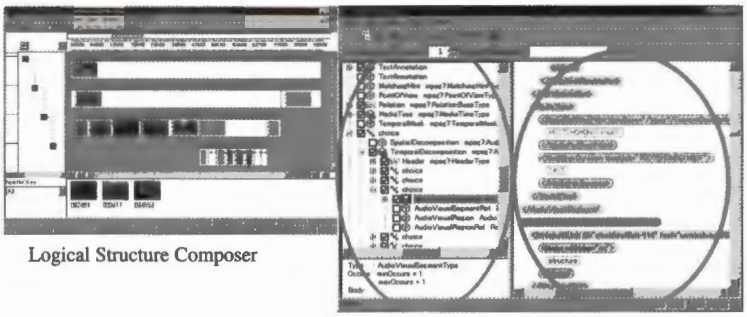


Figure 20.6 Dynamic mapping between structure and MPEG-7 description

Using this segmentation method, the system compares a video frame with the succeeding frame using still image features. If the absolute value for the difference between two such frames exceeds a set threshold value, then the system detects this as a cut-point, that is, there is a shot change in the visual stream. The cut-point is used to signify the start of a new subsegment of the stream. These subsegments are generated automatically where each subsegment can be represented by a thumbnail and is displayed as an index to the segmented content.

The hierarchy structure of the segmented content is mapped dynamically to its equivalent MPEG-7 description and also opens up the MPEG-7 editor window (see also Figure 20.6).

The MPEG-7 description is displayed on the right side and the XML element hierarchy is displayed on the left side of the editor. This view shows not only the MPEG-7 description elements but also the related description scheme candidates that can appear under each element [29].

Users can edit the MPEG-7 description instance manually with the MovieTool by using MPEG-7 tools such as the *UsageInformation Description Scheme* or the *CreationInformation Description Scheme* to describe content in greater detail. Importantly, users can easily see the relationships between the content and its associated MPEG-7 description, a powerful feature since it aids in understanding the usage of the MPEG-7 description tools and how they related to video content.

The current version of the MovieTool is quite generic and includes more than 100 MPEG-7 description tools. Application-specific subsets of tools are currently being developed; see, for example, the 'Aerobatics Video Clip Retrieval System on the Web', and it can be reviewed at *http://ab-movie.ricoh.co.jp*. The purpose of this application is to allow users search for aerobatic aircraft video clips by 'pilot name', type of 'aerobatic maneuver' and 'aircraft' type.

20.9 TV-ANYTIME AND MPEG-7

The TV-Anytime Forum is an international association of organizations, which seeks to develop open specifications to enable audiovisual and other services based on mass-market

high-volume digital local storage in consumer platforms. In particular, these specifications will enable interoperability between PVRs or Personal Digital Recorders, Interactive TV systems and associated services. These services will support searching, selecting, locating and acquiring content wherever and whenever it is made available.

Specifically, the work in the TV-Anytime Forum is organized into the following Working Groups (WG).

1. *Content referencing*: Develops specifications for content identification and acquisition;
2. *Metadata*: Develops specifications for content description;
3. *Content rights management and protection*: Develops specifications for protection and rightful use of content;
4. *Benchmark applications*: Develops usage scenarios and business models;
5. *System description*: Develops end-to-end designs of various model systems.

20.9.1 MPEG-7 Tools Used in TV-Anytime

Cooperation and coordination with MPEG-7 through the work of several companies and individuals active in both standard groups has led to commonalities between the TV-Anytime and MPEG-7 standards.

In particular, the TV-Anytime Metadata specifications are based on an XML/XML Schema framework, identical to that of MPEG-7. Furthermore, the TV-Anytime Metadata specification has adopted a number of complete tools from the MPEG-7 Multimedia Description Schemes specification. In particular, TV-Anytime adopted the User Preferences Description Schemes and the Usage History Description Schemes. The User Preferences Description Schemes define a representation of the preferences, likes and dislikes of users, such as their favorite genre, favorite shows, favorite actors and so forth. The Usage History Description Schemes can be used to describe a history of the user's actions on content, such as viewing or recording a program. Both can be used for personalizing the application or service.

TV-Anytime has developed their own description schemes for describing programs and their groupings (e.g. into series). These description schemes are important to represent information typically found in Electronic Program Guides (EPGs) or Electronic Content Guides (ECGs), such as program title, genre, actors and other technical attributes. The TV-Anytime specification uses various basic types developed within MPEG-7 to build their program information description schemes. Also, TV-Anytime has cooperated with MPEG-7 in building a hierarchical classification scheme that contains terms to express content genres.

Recently, TV-Anytime has been considering the adoption of tools developed by the MPEG-7 Systems group for binarization of textual content descriptions. Currently, both the TV-Anytime Forum and MPEG are developing specifications in the area of Rights Management and Protection. This provides more opportunities for these two standardization groups for cooperation.

More details about TV-Anytime can be found at *http://www.tv-anytime.org*.

20.10 CONCLUSION

The applications listed in this chapter represent only a few of the huge variety of applications that will be possible with MPEG-7 technology. The application ideas may not be new but the benefits of MPEG-7 are that it provides open-standard nonproprietary tools to enable the creation of such applications.

The News content applications from LG Electronic and EPFL/EIF show how, by selecting a combination of tools from the MPEG-7 tool set, similar applications can be created but with different nuances in functionality.

The 'Search by Humming', (MIT), application is a good example of next-generation applications that will come with MPEG-7. This type of content search capability is very much how people would like and expect to be able to do. With CUIDAO, sound studio professionals will have sophisticated tools to enable them search and compare sound samples to help them match or create special effects in the music domain.

The *MovieTool* by Ricoh is significant because it is an MPEG-7 authoring tool with powerful capabilities that show visually the dynamic mapping between rich AV content features and their equivalent description instances in MPEG-7. For MPEG-7 and non-MPEG-7 experts, this capability alone will greatly increase application development speed and make it easier to learn how to use the MPEG-7 tools.

NEC's 'Real-time Video Monitoring System' has already been in operation for more than 12 months and can handle 50 000 video clips over five television channels. One of its key features is the ability to search and find automatically requested icons in video clips – for humans to carry out such a task over 50 000 video clips, it would certainly be a most tiresome task indeed.

In ending this chapter, let us once again consider the study results that show huge volumes of digital information being created each year. Systems for the control and management of this information will be necessary. Fortunately, with MPEG-7, such systems will be enabled.

ACKNOWLEDGMENT

This chapter was written from material prepared by the following groups to whom this author owes many thanks and appreciation. Akio Yamada, *a-yamada@da.jp.nec.com*, *Real-time Video Identification*, NEC, Japan; Youngmoo Kim, *moo@media.mit.edu*, *Query by Humming,* MIT Media Lab, USA; Vincent Puig, *Vincent.Puig@ircam.com*, *CUIDADO,* IRCAM, France; Nastaran Fatemi (*Nastaran.Fatemi@epfl.ch*) and Omar Abou Khaled (*Omar.Aboukhaled@eif.ch*), *Indexing/Retrieval of TV News Programs,* EPFL/EIF, Switzerland; Kyoungro Yoon (*yoonk@LG-Elite.com*) and Jungmin Song (*jmsong73@LG-Elite.com*), *Article-based News Browser,* LG Electronics, Korea; Yuki Wakita, *mm-support@src.ricoh.co.jp*, *MovieTool,* Ricoh Company, Ltd., Japan (MovieTool is developed both by Ricoh Company, Ltd. and Information Broadcasting Laboratories, Inc.); Peter van Beek, *pvanbeek@sharplabs.com*, *TV Anytime Forum,* Sharp Labs, USA.

REFERENCES

[1] P. Lyman and H. R. Varian, How much information, 2000. *http://www.sims.berkeley.edu/how-much-info*.

[2] ISO/IEC 15938-3:2001, "Multimedia content description interface – Part 3: Visual," Version 1.

[3] A. Yamada *et al.*, "Visual program navigation system based on spatial distribution of color", *Proceedings of the IEEE International Conference on Consumer Electronics (ICCE2000)*, Los Angeles, CA, June 2000, pp. 13–15.

[4] E. Kasutani and A. Yamada, "The MPEG-7 color layout descriptor. A compact image feature description for high-speed image/video segment retrieval", *Proceedings of IEEE International Conference Image Processing (ICIP '2001)*, Vol. I, Thessaloniki, Greece, 2001, pp. 674–677.

[5] A. Ghias, J. Logan, D. Chamberlin and B. C. Smith. "Query by Humming: musical information retrieval in an audio database," *Proceedings of the Third ACM International Conference On Multimedia*, San Francisco, 1995, pp. 231–236.

[6] W. J. Dowling, "Scale and contour: Two components of a theory of memory for melodies," *Psychological Review*, **85**(4), 341–354 (1978).

[7] D. J. Levitin, "Memory for musical attributes," in P. R. Cook, ed., *Music, Cognition, and Computerized Sound*, MIT Press, Cambridge, Mass., 1999, pp. 214–215.

[8] R. J. McNab *et al.*, "Toward the digital music library: tune retrieval from acoustic input," *Proceedings of the ACM Digital Libraries*, Bethesda, March 1996, *http://www.nzdl.org*.

[9] MiDiLiB, University of Bonn, *http://leon.cs.uni-bonn.de/forschungprojekte/midilib/english*.

[10] Search by Humming, University of Southampton, *http://audio.ecs.soton.ac.uk/sbh*.

[11] Themefinder™, Stanford University, *http://www.ccarh.org/themefinder*.

[12] TuneServer, University of Karlsruhe, *http://wwwipd.ira.uka.de/tuneserver*.

[13] P. Aigrain, "New applications of content processing of music," *Journal of New Music Research*, **28**(4), 271–280 (1999).

[14] N. J. Belkin, "Helping people find what they don't know," *Communications of the ACM*, **43**(8), 58–61 (2000).

[15] W. W. Cohen and W. Fan, "Web-collaborative filtering: recommending music by crawling the Web," *Proceedings of the Ninth International World Wide Web Conference*, Elsevier, Amsterdam, The Netherlands, 2000, pp. 685–698.

[16] E. Wold, T. Blum, D. Keislar and J. Wheaton, "Content-based classification, search, and retrieval of audio," *IEEE Multimedia*, **3**(3), 27–36 (1996).

[17] F. Gouyon, F. Pachet and O. Delerue, "On the use of zero-crossing rate for an application of classification of percussive sounds," Proceedings of the COST-G6 conference on digital audio effects (DAFx-00), Verona, Italy, December, 2000.

[18] P. Lepain, "Polyphonic pitch extraction from musical signals," *Journal of New Music Research*, **28**, 4 (1999).

[19] F. Pachet, P. Roy and D. Cazaly, "A combinatorial approach to content-based music selection," *IEEE Multimedia*, **7**(1), 44–51, January–March (2000).

[20] F. Pachet and D. Cazaly, A classification of musical genre, Content-Based Multimedia Information Access (RIAO) Conference, Paris, March, 2000.

[21] S. Rossignol *et al.*, Feature extraction and temporal segmentation of acoustic signals, Proceedings of the International Computer Music Conference (ICMC), Ann Arbor, USA, October, 1998.

[22] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *Journal of the Acoustical Society of America*, **103**(1), 588–601 (1998).

[23] ISO/IEC 15938-5:2001, "Multimedia content description interface – Part 3: Multimedia Description Schemes", Version 1.

[24] *http://www.softwareag.com/tamino/*.

[25] "XQuery 1.0: An XML Query Language, W3C Working Draft," June 2001. *http://www.w3.org/TR/xquery/*.

[26] N. Fatemi and O. Abou Khaled, Indexing and retrieval of TV news programs based on MPEG-7, Proceedings of the IEEE International Conference on Consumer Electronics (ICCE '2001), Los Angles, Calif., June, 2001.

[27] N. Fatemi and O. Abou Khaled, "COALA: Content-oriented audiovisual library access," *Proceedings of the 8th International Conference on Multimedia Modeling (MMM '2001)*, Amsterdam, The Netherlands, 2001, pp. 59–71.

[28] K. Yoon *et al.*, "MPEG-7 based news browsing: description extraction, browsing and exchange," *Multimedia Systems and Application IV*, *Proc. of SPIE*, Vol. 4518, Denver, Colo., 2001.

[29] T. Kunieda Y. Wakita, "XML-schema dynamic mapped multimedia content description tool", *Proceedings of 20th International Conference on Conceptual Modeling (ER2001)*, Yokohama, Japan, LNCS 2224, pp. 599–601, 2001.

21

Mobile Applications

Neil Day[1], Shun-ichi Sekiguchi[2] and Mikio Sasaki[3]

[1]MPEG-7 Alliance, [2]NTT DoCoMo, Inc., Japan, [3]DENSO CORPORATION, Japan

21.1 WHAT YOU WANT, WHERE YOU WANT, WHEN YOU WANT. (WYW)[3]

Mobile applications and the distribution of content to wireless devices are some of the most exciting developments taking place in recent times. As we saw in the previous chapter, digital information is growing at a fast pace each year – but it is useless if we cannot access it in convenient and effective ways. This chapter outlines two types of mobile applications – customization and real-time delivery of information for mobile users – that use MPEG-7 technology. They present a picture of promising applications we can expect in the near future and show how MPEG-7 is being used to bring about these applications.

Recently, mobile appliances such as cellular phones, Personal Digital Assistant (PDA) or small laptop PCs with web browsers and mail software have become popular tools for Internet access in both personal and business use. The number of mobile phone subscribers having Internet capability in Japan is now more than 30 million. Moreover, wireless network bandwidth is expected to rapidly widen owing to the advent of the IMT-2000 (*http://www.imt-2000.com/*) communication service.

The first implementation stage of this service will see the offering of broadband packet switched wireless channels of up to 384 kbps. This will enable the delivery of high-quality audiovisual (AV) content (CIF-resolution video with music etc.) to mobile users in the near future. As mobile devices become part of our daily lives in communicating and finding information, the corresponding applications – such as the ones described in this chapter – will help bring digital information to us in meaningful ways on devices wherever we are, whenever we want.

Disclaimer

The applications described here are based on MPEG-7 tools that were defined at the time of writing of this chapter (October 2001) and while every effort has been made to ensure

that the information is up-to-date, there may be exceptions. The main purpose of this chapter is to highlight new application scenarios made possible by MPEG-7.

21.2 CUSTOMIZED MULTIMEDIA CONTENT DELIVERY FOR MOBILE USERS

One of the most important roles of MPEG-7 technology is to provide *hint* or *summary* information for content customization. Today, there exist many different types of terminals that are capable of playing *content*, which include various kinds of media. This being the case, content needs to be customized so that it can be delivered to and played on any type of terminal without having to repeatedly create it to fit each terminal's capability. Another type of content customization is to make a summary of the content taking user's preferences into account. A user-preferred summary is valuable since it represents just what a user wants in terms of content over a long period. There are many issues that need to be addressed in making the mobile content-delivery service a reality. It should be noted that continuous streaming of media-rich content, such as live sports broadcast, to mobile users may be of limited value. A typical mobile device user has little time to enjoy such content! A second issue concerns the conditions involved in networks to be used for content delivery. Most content-delivery systems will be implemented over packet-switched networks, which in general face the quality of service (QoS) [1] control problem due to network congestion or channel errors. It is quite likely that wireless channel conditions will be much more critical than those of fixed-line channels. Thus, a delivery system capable of transmitting only essential content would effectively save network resources and become an instrumental factor in mobile communications.

21.2.1 An Application Scenario

MPEG-7 can play an important role in the construction of an efficient mobile content-delivery system. This section describes a client-driven content customization mechanism using MPEG-7 tools [2]. Figure 21.1 illustrates a scenario for a location-dependent content-delivery system, the concept of which is *'Summary Push and Smart Pull'*. Initially in this scenario, small-sized data representing the *Table Of Contents* (TOC) of a related whole story are pushed to the user, and are used to generate a user-preferred AV summary at the terminal. It is assumed here that the mobile user is already subscribing to one or more service providers.

A typical example of this kind of service is a visual local guide information service including recommended places to go such as restaurants, shops or other spots of interest. The service provider prepares rich AV content (e.g. high-quality MP4 file [3]) and an MPEG-7 instance consisting of scene structure and keyword information of each scene (Japanese restaurant, CD shop, café, bar and so on). The AV content is loaded onto a streaming server. Once a subscribing user enters the service area, a JAR (Java Archive [4]) file is pushed to the user's terminal after an appropriate negotiation process has been completed between terminal and server. This JAR file is called the *Summary Content*, which contains the *MPEG-7 instance*, as mentioned above, a *Java application program* and small *media data* such as thumbnails of key frames (e.g. Joint Photographic Experts Group (JPEG)-encoded images), key sounds or annotation text that represent the TOC.

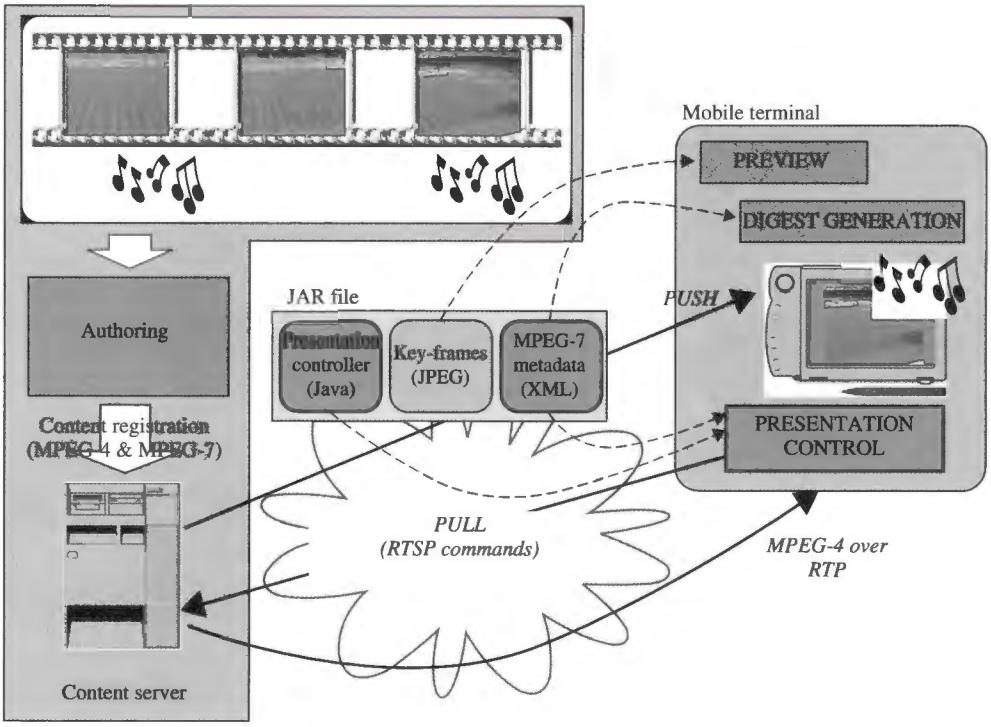


Figure 21.1 System overview

The Java application runs on the terminal and performs automatic generation of the user-preferred AV summary using the MPEG-7 instance and user preference data stored in the terminal. The MPEG-7 *User Preferences* [5] description tools can be used as a format for user preference data. Finally, the AV summary is delivered using streaming technology. The mobile terminal then starts playing a set of AV scenes that represent places where the subscribing user wants to go. The term 'Smart Pull' is derived from the automatic generation of the source of the AV summary streamed content (e.g. pull access to rich AV content).

This application thus enables mobile users to obtain timely AV information depending on their situation (time, location etc.). These users can obtain their customized AV summary automatically.

21.2.2 Usage of MPEG-7 Tools

In the application scenario shown in Figure 21.1, the MPEG-7 instance is utilized by a Java application, which carries out (1) scene filtering and (2) random access to the filtered scene portions of the streaming content (see Figure 21.2).

Scene filtering: Content customization in this scenario essentially corresponds to a scene filtering operation. It consists of matching the keyword information of scenes in an

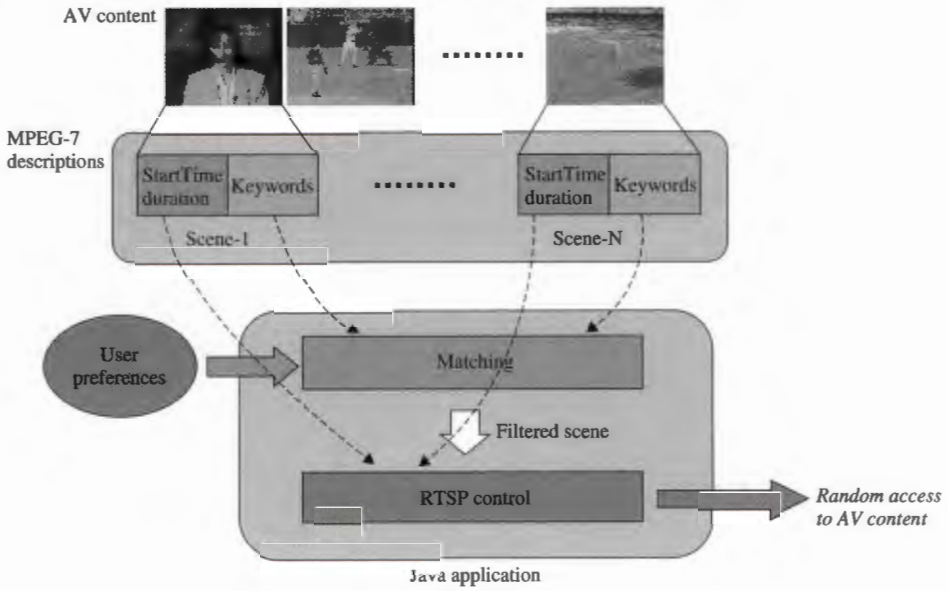


Figure 21.2 MPEG-7 usage scenario

MPEG-7 instance with user-preference data stored in the terminal and selecting a set of appropriate scenes.

Content control: An MPEG-7 instance also provides time information (starting time and duration) of scenes. After the scene-filtering process, the time information of selected scenes is used as the source reference for the content access to form an AV summary.

An example of an MPEG-7 instance used for describing a TOC for soccer news content is given as follows (note that in this example description scheme refers to an MPEG-7 Description Scheme (see Section 3)). Typical keywords used to describe a representative scene in a soccer game include 'goal', 'kick-off' and specific names of the teams or star players. The scene structure is described using the *AudioVisualSegment Description Scheme* (AudioVisualContent). The *MediaInformation Description Scheme* is required to describe basic attributes of the whole content such as *Media Profile* and *Locator* of rich AV content to be delivered. These attributes are used in the streaming process.

Each scene is represented as a child element of *AudioVisualContent*. Each scene description consists of a key-frame locator (TitleMedia), keyword information of that scene (PointOfView) and start time and duration information (MediaTime). The *PointOfView* description is used not only for matching with user-preference data but also as source information to update the user-preference data. Since soccer game keywords can be used for other games as well, they can be stored in the terminal to serve as a source for updating user preferences.

21.2.3 What Still Needs to be Done?

The most critical issue to encourage this kind of application is the availability of authoring tools with MPEG-7 capability. The content structure should be generated



with an intelligent AV scene analysis tools [6] so that the authoring cost including MPEG-7 annotation can be reduced. Such an authoring environment definitely needs to be developed. A second issue is the possibility of extending Mobile Java capability. Since Java is a suitable platform for building Extensible Markup Language (XML) applications and media presentations, it is expected that Java Application Program Interfaces (APIs) for mobile appliances can be further extended to enable them to handle flexible AV content presentations. Content customization in terms of terminal and network resources [7] should also be considered with dynamic summary creation as in the scenario described above.

21.3 REAL-TIME INFORMATION RETRIEVAL FOR MOBILE USERS

One of the most interesting mobile applications for real-time content delivery is the *'In-Vehicle Information System'* [8], which plays a central role in ITS (Intelligent Transportation Systems). Generally, the car navigation system is the most popular system of this type. A total of 6.9 million car navigation system units have been sold in Japan alone as of the first quarter of year 2001 [9]. It is not too much to say that someday all vehicles will have information terminals and cameras. In fact, in the near future, mobile users will not only receive well-defined content but also send real-time information. Then they will be able to receive real-time content generated from massively aggregated information stored on central servers. Therefore, it can be anticipated that equipments such as *in-vehicle* information terminals, onboard AV capture & transmission devices, mobile handy videophones, PDAs and portable PCs will be widely and seamlessly connected to AV terminals at home or at the office for use in daily human information activities.

21.3.1 Content Aggregation and Navigation

Content on demand, for any time and any place, can be gathered by the users' agents. Here the best matching between the provider and the receiver would be performed. Originally in the area of vehicle technology, there has been the concept of the 'Probe Information System' [8], in which every car plays a role of a sensor for a large real-time information gathering system (see Section 2.3 below). This idea also applies for other kinds of mobile terminals. Every mobile user could be a real-time information source in a huge business model of content delivery. In such a system, tremendous amounts of unclassified content is randomly generated and uploaded to a central data warehouse and is navigated on the basis of an indexing system using MPEG-7 metadata descriptions.

21.3.2 Utilization of Dependency

In all of the situations described earlier, some kind of effective filtering methods are desired especially for the mobile user who tends to be in a hurry or for a car driver who should not be distracted by cumbersome tasks of retrieval. Therefore, the utilization of dependency information [10] would be a reasonable solution for the following types of tasks:

- time-dependent tasks (information retrieval for news, events, skiing, swimming etc.);
- location-dependent task (local information, map, guidance to bank, post office, station etc.);
- preference-dependent task (selection of restaurants, shopping, cinema, sports information, broadcast channel etc.);
- situation-dependent task (toll gate information, emergency information, weather information, music information, communication control etc.).

21.3.3 Application Scenario

Figure 21.3 shows the concept of an Agent-based Mobile Content Retrieval System. Although this example describes a scenario that incorporates various technologies that have capture and filtering functions, it does highlight the role MPEG-7 plays in an overall system of information retrieval, that is, content indexed using MPEG-7 metadata [11].

1. An agent in the user's terminal sends the user's situation, demands and query descriptions.
2. At the same time the information captured by the user or the user's terminal is sent to information centers. Information examples include:
   - traffic information (route guidance, congestion, vacancy of parking area etc.);
   - emergency information (disaster, accident, panic etc.);
   - local information (shopping, sightseeing, restaurant, events);
   - information for personal broadcast (hobby, art, news, e-business etc.).

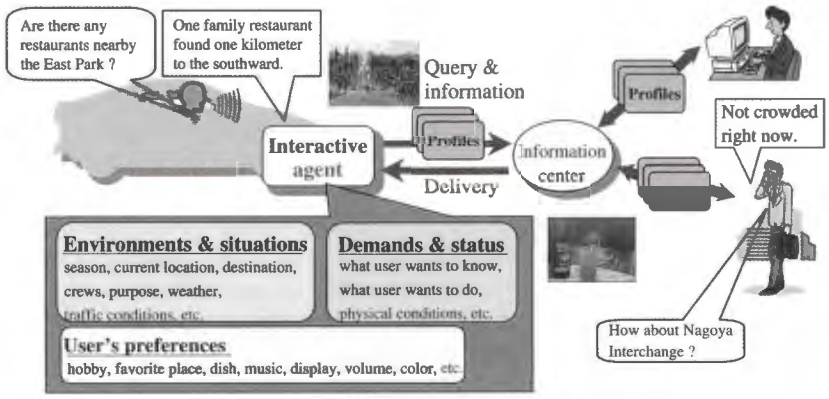


Figure 21.3 Dialogue and contents retrieval system

3. MPEG-7 Metadata corresponding to the above AV information, using situation dependency (time, location, weather, genre etc.), can be retrieved by other users.
4. Agents in the terminals of mobile users in fast vehicles or trains negotiate with the centers to select adaptively and dynamically the optimum time and place and communication system to receive the requested information.

Thus, the users receive the acquired information, which may be a summarization of the AV content. It would be possible for the *in-vehicle* system to read out the summary information of contents so that the system does not distract the driver from driving safely.

21.3.4 What Still Needs to be Done?

MPEG-7 content descriptions may be widely used in the near future over client devices, servers and networks. However, for massively aggregated unclassified content, fast retrieval or prefiltering techniques should be developed. Those techniques will incorporate an integrated description framework of user environments. For specialized environments in which the application context is important, MPEG-7 will provide the basis for associating the correct semantic description of the meaning of content with its context.

Content descriptions

The usage of MPEG-7 tools to describe content must be clear. A developer must not be confused over which tools are appropriate when building an application. This is potentially a problem when trying to decide the optimum toolset to use and has implications for interoperability of applications and tools [2]. As the MPEG-7 standard is fixed, it will be necessary to carry out extensive testing of its tools as used in real-world applications to understand if they have been used according to how they were intended.

Query description

Study of the nature of queries posed by users will be necessary. Although the benefits of MPEG-7 will provide users with more rich responses (audio and visual content) to their queries, it will be necessary to evaluate the new user query-models that MPEG-7 will enable, to understand how to create optimum application-specific search and browse interfaces MPEG-7's extensibility to allow the standardization of new features of content, is expected to be utilized as the demand for new content features grows.

21.4 CONCLUSION

What type of killer applications will MPEG-7 enable? This was one of the questions asked at a panel session during the second MPEG-7 awareness event in Singapore (March 2001, *www.mpeg-industry.com*). The consensus then was that a strong candidate can be found in the domain of mobile applications, which are created in combination from MPEG-7 tools for navigation and MPEG-4 tools for AV Object coding. Although MPEG-7 is a

new standard, we can expect a rapid uptake of technology solutions that it can provide, especially in the world of mobile devices. MPEG-7 will certainly impact on our daily lives through the little screens of our mobile devices and via our car navigation systems, eventually. Not only will we benefit from the delivery of content where we want, when we want, but delivery systems will tend to become more 'intelligent'. Knowing the context of content and the user's location and preferences will ensure, for example, that the delivery of content is localized as we move from one environment to another. We live in exciting times indeed.

MPEG-7 not in Isolation:

MPEG-7 provides solutions for describing AV content. However, such solutions are not supposed to live in isolation but instead will be used in various types of information services and with other standards. In delivering MPEG-7-described content between server and client, there are various impacting technologies that need to be taken into account. For example,

1. Transcoding such as MPEG2 to MPEG4 will be required if broadcast content of high quality is the aim for mobile retrievals.
2. Mobile-related user environments (time, place, velocity, cost, bandwidth, e.g. using MPEG-21) would be described so that media conversion (transcoding, partition, synthesis, feature extraction, text to speech etc.) methods and communication channels can be controlled. Some user environments are already embodied in handy-phone systems.
3. Capability exchange (display size, memory, CPU power etc.) will be required to realize the interoperability between different kinds of terminals such as portable phones, car navigation systems, PDAs and PCs, [12].

In conclusion, we anticipate MPEG-7 to have a significant impact on the future of information processing in mobile environments.

ACKNOWLEDGMENTS

The authors would like to thank DENSO CORPORATION and NTT DoCoMo Inc., for their support and permitting them to describe their MPEG-7 work. Shun-Ichi Sekiguchi is now with Mitsubishi Electric, Japan.

REFERENCES

[1] P. Ferguson and G. Huston, *Quality of Service*, John Wiley & Sons, New York, 1998.
[2] MPEG Requirements Group, "Study of MPEG-7 Mobile Requirements & Applications V.2," ISO/IEC/JTC1/SC29/WG11/N4036.*http://mpeg.telecomitalialab.com/working_documents.htm #MPEG-7*.
[3] ISO/IEC 14496-1:2001, "Information technology–Coding of audio-visual objects – Part 1: Systems," October 2001.
[4] JavaTM Archive Features, *http://java.sun.com/j2se/1.3/docs/guide/jar/index.html*.
[5] ISO/IEC 15938-5:2001, "*Multimedia Content Description Interface – Part 5: Multimedia Description Schemes*," Version 1.

[6] R. Brunelli *et al.*, "A Survey on the Automatic Indexing of Video Data", *Journal of Visual Communication and Image Representation*, **10**, 78–112 (1999).

[7] M. Nilsson, J. Hjelm and H. Ohto, Composite Capabilities/Preference Profiles: Requirements and Architecture, W3C Working Draft, July 2000. *http://www.w3.org/TR/2000/WD-CCPP-ra-20000721/*.

[8] *http://www.ipcar.org/english/e_index.html*.

[9] *http://625.uk.com/digital/*.

[10] M. Sasaki and F. Murase, The Latest Evaluation of Dependency Description Scheme, Technical Report #21-01, March 2001.

[11] M. Sasaki and F. Murase, MPEG-7 Instance for Mobile Content Delivery, Technical Report #21-03, September 2001.

[12] M. Sasaki, H. Nanba and K. Matsugatani, Adaptive Communication Using Metadata, Technical Report #21-02, October 2000.



This document is available on the DVD

Index

3-D
  mesh, 248, 254
  models, 9
  objects, 231
  shape descriptor, 180
  Shape Spectrum Descriptor (SSD), 231

Absolute Context Mode, 62
Abstract, 147
Abstraction model, 127
Abstraction, 127
AbstractionLevel, 133
Access Units (AUs), 34–5
AddContent, 64
Adjacent notes, 296
Aerial image, 213, 218, 228
Affective description, 96, 100
Affective response, 95, 101, 107
  Agent description scheme, 132
  agent object, 112
  Agentobject description scheme, 130
Agents
  agents, 96, 106, 112
Aggregation, 12
Agglomerative clustering, 221
Alexandria Digital Library, 218
Amendment (AMD), 14
Amplitude envelope, 22
Analytic clips/transitions, 118
Analytic model, 154
AnalyticClip description scheme, 117
AnalyticClip, 117
AnalyticTransition description scheme, 118
AnalyticTransition, 118
Angular Radial Transform (ART), 233, 238
Anonymous type, 52
Architecture, 33
ART, 238–40
ASR
  ASR Automatic speech recognition (ASR) systems, 299
  system, 303
  technology, 301
  training set, 303
Atomic, 48
Attributegroup, 53
Audio
  audio, 13, 98, 283
  fingerprinting, 291
  segment, 287
  spectrum envelope, 319
  description framework, 286
  description schemes, 95
  descriptors, 286
  editing, 285
  editor, 285
  features, 286
  LLD scalartype, 287
  LLD vectortype, 287
  preprocessing, 302
  QBE, 328
  query-by-example, 317
  sampling frequencies, 287
  segment, 118
  signal, 286
  signal processing, 295, 317
  signature tools, 286
  standard, 283
  stream, 295
  audio-visual regions, 118
  audio-visual segment, 118
  AudioDSType, 302
  AudioHarmonicityType, 292
  AudioPowerType, 289
  AudioSegment, 287

AudioSegment description scheme, 117
AudioSourceLocator, 143
AudioSpectrumBasis, 317
AudioSpectrumCentroid, 287
AudioSpectrumCentroidType, 291
AudioSpectrumEnvelopeType, 289
AudioSpectrumProjection, 317
AudioSpectrumSpreadType, 291
AudioSummaryComponent, 143
AudioWaveformType, 289
automatic audio classification, 326–27
AudioSegment, 117
Audiovisual
audiovisual, 3
conceptual model, 12
content, 299
documents, 111
sequences, 111
AudioVisualRegion description scheme, 117
AudioVisualSourceLocator, 143
AudioVisualRegion, 117
AudioVisualSegment, 117
Author-generated information, 113
Availability description scheme, 116
Availability, 116
Average Normalized Modified Retrieval Rate (ANMRR), 181, 184, 188, 203
Average Rank (AVR), 184

BaseRelation, 135
Basic data types, 96
Basic elements, 84, 95
BasicDuration, 59
BasicTimePoint, 59
Basis functions, 318, 325
Basis vectors, 181
Bayesian methods, 172
Bijective, 204
BiM, 35, 61
Bin unification, 207
Binarization, 13, 313
Binary format, 61
Binary payload, 72
Brodatz album, 218, 221
Browsing, 139
Browsing, navigation, visualization, 87
BrowsingPreferences description scheme, 169–70
BrowsingPreferences, 170
Bull's-Eye Percentage (BEP), 237

Camera motion, 179–81, 261–2, 266, 270
Cartoon search, 256
Choice, 49
CIE LUV, 195
Classification, 218, 223
classification description scheme, 116
classification model, 154, 159
classification model tool, 159
classification schemes, 96, 100, 104
classification schemes and terms, 100, 104
ClassificationPreferences description scheme, 169
classifier, 327
ClassificationPreferences, 170
Cluster model, 154
COALA, 345
Codec, 63
Coding Schemes, 83
Collection description tools, 111
Collection model tool, 158
Collection tools, 154
Collection, 88, 154
Color
color, 8, 180, 204
descriptors, 180, 187
distribution, 180
features, 180
layout, 338
Layout Descriptor (CLD), 338
Queries (CCQ), 188
space descriptor, 188
spaces, 189
structure descriptor, 188
structure histogram, 205
variance, 197
Common Color Dataset (CCD), 188
dominant color, 212, 258
dominant color descriptor, 188
dominant colors, 196
HMMD color space, 188, 191–2
scalable color descriptor, 188
Committee draft, 14
Complex, 47
Complexcontent, 49–50
Complextype, 49
ComponentSourceTime, 143
Concept description scheme, 131
Concept, 111, 131
Conceptual aspects, 87
Conceptual model, 12

Conceptual notions, 87
Conformance testing, 14
Connected components. connectivity, 118
Consumers, 164
Consumption, 165
Content
  content browsing engine, 168
  content control, 356
  content creation, 115
  content creators, 169
  content description, 3
  content filter, 168
  content management, 83, 86, 112
  content organization, 88, 153
  content providers, 164
  content referencing, 349
  content semantics, 126
  content structure, 116
  content usage, 116, 163
ContentAbstraction, 90
ContentEntity, 89
ContentManagement, 92
Context
  mode, 62, 65
  node, 61–2
  path, 62
Contour, 243
Contour-shape, 234
Coordinated universal UTC, 167
Cor anglais, 289
Core experiment, 212
Corel, 218
Creation description scheme, 115
Creation information, 86, 112, 114
Creation location, 112
Creation, 112–13
CreationInformation description scheme, 130
CreationPreferences scheme, 169
CreationPreferences, 169
Cross-modality, 17
CUIDADO, 342
Curvature function, 243

Decomposition, 88, 144
DeleteContent, 64
Dependency structure, 103
Derived, 47
Description
  Definition Language (DDL), 11, 13, 35, 43, 83
  metadata, 85
  ordering, 95
  scheme, 4, 97
  schemes, 83
  systems, 8
  tree, 37, 61
  vocabularies, 95
Description, 10
Descriptor collection tool, 156
Descriptor, 12, 97
Descriptors, 83
Dialect adaptation, 305
Digital and analog multimedia content, 111
Discrete Cosine Transform (DCT), 188
Dissimilarity, 240
Dominant scale, 219
DSType, 85
DType, 85
Dynamic time warping (DTW), 325

Eccentricity, 236
Editing work, 118
Edge features, 223
Eigenspectra, 318
Eigenvalues, 182
Eigenvectors, 181–2
Electronic program guide, 166
Element, 46
Elisions, 303
Emotional, 107
Encoder, 63
Entity, 12
Enumeration, 48
Euclidean distance, 182, 196
Event description schemes, 130
Events, 111, 126
eXperimentation Model (XM), 10
Explanatory metadata header, 299
Explanatory metadata, 306
Expression-based audio retrieval, 303
Extended Entity Relation (EER), 127
Extensibility, 10
Extensible Markup Language (XML), 9, 35, 61, 97, 335
Extension, 50, 77
Extraction engine, 285

Face descriptor, 181
Face recognition, 181
Facets, 56
File format, 115

Final Committee Draft stage (FCD), 14
Final Draft Amendment (FDAM), 14
Final Draft International Standard (FDIS), 14
Final Proposed Draft Amendment (FPDAM), 14
Financial data, 116
Finite State Model, 158
Finite-state automaton decoders, 74
Finite-state model types, 157
Formal abstraction, 128
Forward/Backward compatibility, 77
Fourier transform, 216–17
Fragment Update command, 61
Fragment Update context, 61
Fragment Update payload, 61
Free term use, 105
Free text, 102, 143
Frequency, 145
Frequency plane, 214, 216
Fuzzy clustering, 172

Gabor functions, 215–16
Gaps, 128
Gaussian distribution, 322
Gaussian, 157
Gazetteer, 219
Genre, 86
Global-edge histogram, 224–5, 227
Grammar, 103
Graph description scheme, 101
Graph, 100
Graphical classification schemes, 106
Group of frames, 188
Group, 53

Haar transform, 198
Hamming distance, 200
Hamming window, 294
Harmonic ratio, 292
Header, 90
Hidden Markov Model (HMM), 158, 304
Hierarchical decompositions, 116
Hierarchical navigation, 141
Hierarchical taxonomy, 297
HierarchicalSummary, 141
Hierarchy, 88
High-level features, 83
Highlights, 148
HMM, 158
href, 97

idref, 97
ImageLocator, 97
Image matching, 224
Implicit relationships, 137
Incremental time, 99
Independent Component Analysis (ICA), 295, 319
Indexing tools, 286
Inflection points, 242
Ink content, 111
Ink segment, 118
InkSegment, 117
Inline term use, 105
International Standard (IS), 14
Interoperability, 8–10, 187, 212
InverseRelation, 125
ISO 8601 standard, 98
ISO 8601, 98
iso-color planes, 204

Joint Photographic Experts Group (JPEG), 354
JTC, 4

Karhunen-Loeve, 185
Key audio segments, 140
Key frames, 140
Keyword, 102, 169

Lambda abstraction, 128
Linear programming, 325
Linking, 95
List, 48
LocationDescription, 115
Local-edge distribution, 223
Log attack time descriptor, 296
Logarithmic spectral description, 292
Loudness, 285, 296
Low-level audiovisual features, 83
l_1 distance, 225

Management, 111
Markov chains, 157
Master media, 86
Master, 113
MatchingHint, 120
Matrix, 96
maxOccurs, 46
Media, 113
  abstraction, 128
  authoring, 179
  description, 86

format, 112, 170
information, 112, 114
instance, 112
localization, 95
localization, annotation, 95
locators, 96–7
profile, 87, 112
quality, 113
review, 116
time, 98, 167
time periods, 98
media-locator, 285
mediaDuration data type, 98
MediaFormat descriptor, 115
MediaInformation description scheme, 114
MediaInstance description scheme, 115
MediaLocator, 97
MediaOccurrence element, 133
MediaQuality descriptor, 115
MediaSourceDecomposition description schemes, 121
MediaSpaceMask, 120
mediaTimePoint data type, 98
MediaTranscodingHints descriptor, 115
MediaFormat, 115
MediaIdentification, 115
MediaInstance, 115
MediaLocator, 97, 115
MediaOccurrence, 133
MediaProfile, 115
MediaQuality, 115
MediaTranscodingHints, 115
Melody, 8
melodic similarity matching, 296
contour description scheme, 296
contour representation, 296
description, 296
description tools, 286, 296
representation, 296
sequence description scheme, 296
Message annotation, 284
minOccurs, 46
Mobile applications, 353
Modalities, 88
Model, 154
inference, 324
tools, 154
Monophonic melodic information, 296
Mood, 107, 284
Mosaic description scheme, 117
Mosaic or panoramic description scheme, 117
Mosaic, 117–18
Motion activity, 262, 266–7
Motion descriptors, 180, 261
Motion trajectory, 261–2, 273
Movie tool, 347, 350
Moving Picture Experts Group (MPEG), 3
Moving region, 118
MovingRegion, 117
MPEG, 8
MPEG-4, 117
MPEG-7 awareness event, 359
MPEG-7 schema, 43
MPEG-7 XM, 11
Mpeg7 root, 84
Multimedia, 111
Multiple Payload Mode, 66
Multiple-view descriptor, 234, 247
Music browser, 342
Music genres, 317
Musical genre, 284, 297

Namespaces, 45
Narrative worlds, 111, 127, 129
Narrow-band spectral information, 318
Navigation, 87, 139
NMTOKENS, IDREFS, 57
Nondirectional, 223–4
Nonedge block, 223–4
Nonnormative, 10
Nonrigid deformations, 236
Normalized autocorrelation function, 221
Normative tools, 13

Object description scheme, 130
Object motion, 181
Object size, 180
Objective measure of retrieval effectiveness, 183
Objects, people and places, 126
Octave bandwidth spectrum, 318
Ontologies
ontologies, 19
sound recognizers, 295
Optical flow, 263
Ordering descriptions, 108
Ordering key, 96
Ordering tools, 100
OrderingKey, 108

Organization, 107
Orientable mesh, 253
Original (source) content, 140
Out-of-vocabulary (OOV) errors, 303
Out-of-vocabulary (OOV) words, 295
Overlaps, 122

Packaging, 85
Parametric motion, 261–2, 276
Parental guidance, 116, 170
Parser, 43
Partition, 88, 144
Payload, 63
People, 95, 100
Perceptual, 214, 219
  features, 296
  sharpness, 291
Percussive instrument descriptor, 296
Percussive, nonsustained sounds, 296
Person identification, 309
Person, 106
Personal preferences, 163
Personalization, 163
PersonGroup, 106
Phone lexicon, 308
Phone links, 302
Phoneme, 302
Phonetic decodings, 303
Phonetic representation, 284
Pitch interval, 297
Pitch, 292, 296
Place description scheme, 132
Places, 95, 100, 107
PointOfView descriptors, 120
PointOfView, 120
Polar frequency, 216–17
Polymorphism, 70
Position code, 68, 71
Preference value attribute, 170
Preferences, 163
Previews, 148
Primitive data types, 54
Principal component analysis (PCA), 181
Principal curvature, 248
Privacy, 168
Probability, 96
  distribution, 157
  model, 154
Profile, 113
Projection slice theorem, 217
Projection, 217, 219–21, 228
Proposed Draft Amendment (PDAM), 14
Pull applications, 16
Push applications, 16

Quad-tree, 145
Quality of Service (QoS), 23, 354
Query by humming, 296
Query paradigm, 284
Query sound, 327
Query target, 285
Query-by-example (QBE), 183, 188, 203, 228, 231, 283, 339
Query-by-sketch, 227

Radial index, 215
Radon transform, 217, 220
Rank, 183
Raster scan, 182
Rate control, 10
Real-world semantics, 87
Recognition engine, 284
Recognition performance, 327
Recognition rate, 303
Reference software, 11, 14
Region and contour descriptors, 180
Region-based Shape Descriptor, 232
RelatedMaterial description scheme, 116
RelatedMaterial description, 115
Relation, 100–1
  description scheme, 101
  tools, 100
Relative Context Mode, 62
Relative time, 99
Repetitive pattern, 219
ReplaceContent, 64
Requirements, 9, 16, 34
Reset, 64
Restriction, 50
Retrieval accuracy, 184, 212
Retrieval of words spoken, 300
Retrieval performance, 7
Rhythmic information, 297
Richness, 296
Rights, 86, 112–13
  data, 116
  holders, 86
Rotation-invariant, 218

Scalability, 187, 199
Scene filtering, 355
SceneGraph, 120
Schema Branch Code (SBC)
Schema, 23

ID, 64
language, 44
tools, 83, 85, 95
Scripted speech, 300
Search engines, 7, 180
Segment, 87, 111, 149
attributes, 120
decompositions, 120–21
entities, 117
multimedia description scheme, 287
type, 287
Semantics, 111, 219, 223–4
Semantic attributes, 132
Semantic entities, 111, 129
Semantic labels, 154
Semantic relation, 112, 134
Semantic searching, 303
SemanticBase, 132
SemanticPlace, 130–1
SemanticTime, 130
Semiglobal-edge histogram, 225
Sequence, 49
Sequential navigation, 141
SequentialSummary, 141
Service providers, 164
Shape clusters, 235
Shape descriptors, 180, 231
Shape features, 231
Shape index, 253
Shape of image objects, 180
Signal envelope, 292
Signal, 144
Silence descriptor, 295
Similarity matching, 196
Similarity measure, 180–1, 240
Simple time, 99
Simple, 47
simpleContent, 49–50
Singular value decomposition (SVD), 295, 318
Slideshow, 141
SMIL, 98
Software agents, 164
Sonification, 87
Sound
classification, 317
effects, 317
index database, 328
palette, 343
phenomena, 317
probability models, 322
recognition, 286
recognition tools, 317
similarity, 317
textures, 325
types, 318
SoundClassificationModel, 317, 320
SoundLocator, 143
SoundModel, 323
SoundModel Description Schemes, 322, 327
SoundSpotter, 328
Source program, 147
Source, 101, 170
SourceID, 142
SourceInformation, 142
SourcePreferences description scheme, 169–70
SourcePreferences, 170
Spatial and temporal segments, 111
Spatial coherence, 197, 198
Spatial layout, 180
Spatial relation, 112
Spatial segment decomposition, 112
Spatial structure, 180
SpatialMask, 120
SpatialRelation classification scheme (CS), 124–5, 135
SpatialRelation, 124–5
SpatialSegmentDecomposition, 121
SpatioTemporal segmentdecomposition, 121
SpatioTemporalMask descriptors, 120
SpatioTemporalMask, 120
Speaker information, 309
SpeakerInfoType, 302
Spectral basis representations, 287
Spectral centroid descriptor, 296
Spectral centroid, 285
Spectral envelope, 285
Spectral timbral, 286
SpectralCentroid, 294, 296
Speech and nonspeech classifier, 327
Speech recognition, 284, 304
Speech retrieval, 283
Speech sounds, 327
Spoken annotation, 300
Spoken content, 284
description, 295
metadata, 301
tools, 286, 295
Spoken document retrieval (SDR), 314
Spoken phrase, 301

SpokenContent description scheme (DS), 299
SpokenContent lattice, 310
SpokenContentDS, 300–1
SpokenContentHeader, 306
SpokenContentLattice, 306, 310
Standard Generalized Markup Language (SGML), 9
Start time point, 98
Starting note, 296
Still region, 118
StillRegion, 117
StillRegion3D, 117
Stochastic matrix, 323
Storage, 13
Stream management, 34
StreamID, 98
Streaming audio, 283
Strings, 96
Structural aspects, 87
Structural relations, 124
Structural relationships, 116
Structure, 111
Structured annotations, 103
Subject, genre, 116
Subsegments, 119
Substitution code, 68–9
Summaries, 87
Summarization tools, 139
Summarization, 139
Summary theme, 170
Summary, 140
SummarySegmentGroup, 142
SummaryThemeList, 142
Surveillance, 19, 181, 262
Synchronized Multimedia Integration Language (SMIL), 98
Synchronized, 143
SyncTime element, 143
Synopses, 148
Systems, 13, 33
  architecture, 33–4
  tools, 11, 19

Table of contents, 121, 148
TAMINO, 344–5
Target, 101
Taxonomy of sound classes, 295
Teletype captioning, 299
TeM, 35, 61
Temporal evolution, 325
Temporal multimedia, 98
Temporal timbral descriptors, 292
Temporal timbral, 286
Temporal-Mask, 120
TemporalRelation CS, 124–5, 135
TemporalRelation, 125
TemporalSegmentDecomposition, 121
TemporalSegmentLocator, 97
Term reference use, 105
Terminal architecture, 35
Terms, 100
Text annotation, 95, 102, 284
Textual annotations, 100
TextualSummaryComponent, 143
Texture descriptors, 180
Texture, 8, 180
  descriptor, 213–228
  image texture, 213, 215
  textons, 213
  Julesz, 213
  browsing, 213–14, 219, 221, 228
  similarity retrieval, 213–14
  homogeneous texture, 213–15, 217, 221
  homogeneous texture descriptor (HTD), 214–19
  edge histogram descriptor (EHD), 213–14, 223–7
  regularity, 214, 219–21, 223
  coarseness, 214, 219–20, 223
  directionality , 214, 219–20, 222–3
  texture browsing descriptor (TBD), 219–23
Timbral spectral LLD, 296
Timbre descriptors, 296
Timbre, 22, 284
Time, 98
  data type, 132
  periods, 98
  zone, 98
Title of the content, 169
Title, 297
Top-level elements, 84
Trademark images, 228
Trademark retrieval, 258
Trademarks, 180, 236, 258
Transcoding, 115, 360
Transport streams, 36
Transport, 13
Tune, 284
TV-anytime, 348–9
Type code, 68
Type codes, 70

Unified Modeling Language (UML), 39, 43, 166
Uniform Resource Identifier (URI), 97
Uniform Resource Identifiers, 38
Uniform silence segment, 286
Union Data Type, 57
Union, 48
Unique identifiers, 97
UniqueID, 97
Universal Resource Name (URN), 142
URI, 97
URL, 284
Usage, 111, 113
  history, 20–1, 88, 165
  history description, 163
  UsageRecord description scheme, 116
  patterns, 163
UsageRecord, 116
  User, 163
  authentication, 181
  interaction, 88, 163
  preference descriptions, 166
  preferences, 88, 168–9, 355
  preferences description scheme, 165
  profiling engine, 166
  terminal, 163
  UserAction description scheme, 167
  UserActionHistory description scheme, 167
  UserActionList description scheme, 167
  users' history, 163
  users' preferences, 163

Valid descriptions, 89
Variation program, 147
Variation set, 147
Variations, 88, 139, 146
Vector quantization, 212
Vector, 96
Vibrato modulation, 294
Video
  browsing, 261, 269
  hyperlinking, 262
  identification, 338
  indexing, 261
  objects, 179
  repurposing, 262
  segment, 118
  summarization, 262
VideoSegment description scheme, 117
VideoSegment, 117
VideoSourceLocator, 143
VideoText description scheme, 117
VideoText, 117
View, 88, 139
  decomposition, 144–5
  tools, 139
  tree, 144
Virage Videologger, 8
Visual description schemes, 95
Visual descriptors, 4, 179
VisualSummaryComponent, 143
Viterbi algorithm, 327
Vocabularies, 104
Voice-based query, 284
VRML (Virtual Reality Modelling Language), 254

W3C, 44
Wavelet, 145
WordIndexType, 302
WordLexiconType, 310
WordType, 310
Working Draft, 14
Working group (WG), 10
Wrapper, 45
Wrappers, 89

XM, 11
XML Schema Language, 17
XML Schema, 43
XML, 35, 43
XPath, 97
xpath, 97
XQL, 345
XSL, 345

Year of recording, 297

Zernike moments, 241

Introduction to MPEG-7

MULTIMEDIA CONTENT DESCRIPTION INTERFACE

Edited by
B. S. Manjunath
University of California, Santa Barbara, USA

Philippe Salembier
Universitat Politècnica de Catalunya, Barcelona, Spain

Thomas Sikora
Heinrich-Hertz-Institute (HHI), Berlin, Germany

The use of digital cameras, personal computers and the Internet has made producing and distributing multimedia content easier today than ever before. The new ISO/MPEG-7 standard enables navigation through this wealth of audio-visual content and is set to revolutionise multimedia technology. *Introduction to MPEG-7* takes a systematic approach to the standard and provides a unique overview of the principles and concepts behind audio-visual indexing, metadata description, information retrieval and browsing.

Essential reading for engineers building MPEG-7 compliant systems, the tutorial approach provides an invaluable tool for developers within the multimedia industry and the research and development community. Graduate students and those new to the field will benefit from the overview of search and retrieval technology.

Featuring:

- Practical guidance to the new ISO/MPEG-7 standard written by the experts who have actively participated in its development.
- Clear explanations of the standard's underlying tools, technologies and applications.
- Coverage ranging from the Description Definition Language and systems architecture through to low-level media descriptors.
- Comparisons of related multimedia description and indexing standards.
- Accompanying DVD featuring MPEG-7 reference software along with working demonstrations of the standard and its components.

ISBN 0-471-48678-7
9 780471 486787

WILEY
wiley.com

Manjunath
Salembier
Sikora
Editors
Introduction to MPEG-7
WILEY