

Overview of the H.264/AVC Video Coding Standard

Thomas Wiegand, Gary J. Sullivan, *Senior Member, IEEE*, Gisle Bjøntegaard, and Ajay Luthra, *Senior Member, IEEE*

Abstract—H.264/AVC is newest video coding standard of the ITU-T Video Coding Experts Group and the ISO/IEC Moving Picture Experts Group. The main goals of the H.264/AVC standardization effort have been enhanced compression performance and provision of a “network-friendly” video representation addressing “conversational” (video telephony) and “nonconversational” (storage, broadcast, or streaming) applications. H.264/AVC has achieved a significant improvement in rate-distortion efficiency relative to existing standards. This article provides an overview of the technical features of H.264/AVC, describes profiles and applications for the standard, and outlines the history of the standardization process.

Index Terms—AVC, H.263, H.264, JVT, MPEG-2, MPEG-4, standards, video.

I. INTRODUCTION

H.264/AVC is the newest international video coding standard [1]. By the time of this publication, it is expected to have been approved by ITU-T as Recommendation H.264 and by ISO/IEC as International Standard 14 496–10 (MPEG-4 part 10) Advanced Video Coding (AVC).

The MPEG-2 video coding standard (also known as ITU-T H.262) [2], which was developed about ten years ago primarily as an extension of prior MPEG-1 video capability with support of interlaced video coding, was an enabling technology for digital television systems worldwide. It is widely used for the transmission of standard definition (SD) and high definition (HD) TV signals over satellite, cable, and terrestrial emission and the storage of high-quality SD video signals onto DVDs.

However, an increasing number of services and growing popularity of high definition TV are creating greater needs for higher coding efficiency. Moreover, other transmission media such as Cable Modem, xDSL, or UMTS offer much lower data rates than broadcast channels, and enhanced coding efficiency can enable the transmission of more video channels or higher quality video representations within existing digital transmission capacities.

Video coding for telecommunication applications has evolved through the development of the ITU-T H.261, H.262 (MPEG-2), and H.263 video coding standards (and later enhancements of H.263 known as H.263+ and H.263++),

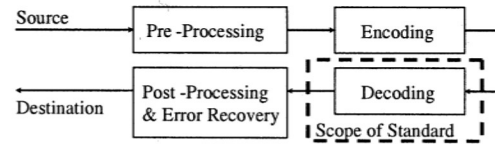


Fig. 1. Scope of video coding standardization.

and has diversified from ISDN and T1/E1 service to embrace PSTN, mobile wireless networks, and LAN/Internet network delivery. Throughout this evolution, continued efforts have been made to maximize coding efficiency while dealing with the diversification of network types and their characteristic formatting and loss/error robustness requirements.

Recently the MPEG-4 Visual (MPEG-4 part 2) standard [5] has also begun to emerge in use in some application domains of the prior coding standards. It has provided video shape coding capability, and has similarly worked toward broadening the range of environments for digital video use.

In early 1998, the *Video Coding Experts Group* (VCEG) ITU-T SG16 Q.6 issued a call for proposals on a project called H.26L, with the target to double the coding efficiency (which means halving the bit rate necessary for a given level of fidelity) in comparison to any other existing video coding standards for a broad variety of applications. The first draft design for that new standard was adopted in October of 1999. In December of 2001, VCEG and the *Moving Picture Experts Group* (MPEG) ISO/IEC JTC 1/SC 29/WG 11 formed a *Joint Video Team* (JVT), with the charter to finalize the draft new video coding standard for formal approval submission as H.264/AVC [1] in March 2003.

The scope of the standardization is illustrated in Fig. 1, which shows the typical video coding/decoding chain (excluding the transport or storage of the video signal). As has been the case for all ITU-T and ISO/IEC video coding standards, only the central decoder is standardized, by imposing restrictions on the bitstream and syntax, and defining the decoding process of the syntax elements such that every decoder conforming to the standard will produce similar output when given an encoded bitstream that conforms to the constraints of the standard. This limitation of the scope of the standard permits maximal freedom to optimize implementations in a manner appropriate to specific applications (balancing compression quality, implementation cost, time to market, etc.). However, it provides no guarantees of end-to-end reproduction quality, as it allows even crude encoding techniques to be considered conforming.

This paper is organized as follows. Section II provides a high-level overview of H.264/AVC applications and highlights some key technical features of the design that enable improved operation for this broad variety of applications. Section III explains the network abstraction layer (NAL) and the overall structure

Manuscript received April 15, 2002; revised May 10, 2003.

T. Wiegand is with the Fraunhofer-Institute for Telecommunications, Heinrich-Hertz-Institute, Einsteinufer 37, 10587 Berlin, Germany (e-mail: wiegand@hhi.de).

G. J. Sullivan is with the Microsoft Corporation, Redmond, WA 98052 USA (e-mail: garysull@microsoft.com).

G. Bjøntegaard is with the Tandberg, N-1324 Lysaker, Norway (e-mail: gbj@tandberg.no).

A. Luthra is with the Broadband Communications Sector, Motorola, Inc., San Diego, CA 92121 USA. (e-mail: aluthra@motorola.com)

Digital Object Identifier 10.1109/TCSVT.2003.815165

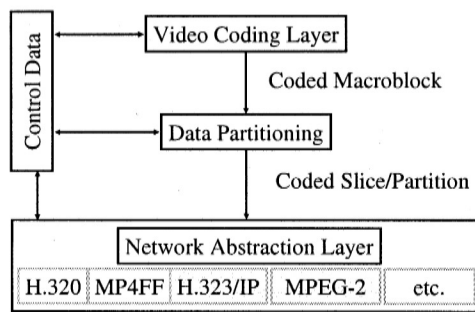


Fig. 2. Structure of H.264/AVC video encoder.

of H.264/AVC coded video data. The video coding layer (VCL) is described in Section IV. Section V explains the profiles supported by H.264/AVC and some potential application areas of the standard.

II. APPLICATIONS AND DESIGN FEATURE HIGHLIGHTS

The new standard is designed for technical solutions including at least the following application areas

- Broadcast over cable, satellite, cable modem, DSL, terrestrial, etc.
- Interactive or serial storage on optical and magnetic devices, DVD, etc.
- Conversational services over ISDN, Ethernet, LAN, DSL, wireless and mobile networks, modems, etc. or mixtures of these.
- Video-on-demand or multimedia streaming services over ISDN, cable modem, DSL, LAN, wireless networks, etc.
- Multimedia messaging services (MMS) over ISDN, DSL, ethernet, LAN, wireless and mobile networks, etc.

Moreover, new applications may be deployed over existing and future networks. This raises the question about how to handle this variety of applications and networks.

To address this need for flexibility and customizability, the H.264/AVC design covers a VCL, which is designed to efficiently represent the video content, and a NAL, which formats the VCL representation of the video and provides header information in a manner appropriate for conveyance by a variety of transport layers or storage media (see Fig. 2).

Relative to prior video coding methods, as exemplified by MPEG-2 video, some highlighted features of the design that enable enhanced coding efficiency include the following enhancements of the ability to predict the values of the content of a picture to be encoded.

- **Variable block-size motion compensation with small block sizes:** This standard supports more flexibility in the selection of motion compensation block sizes and shapes than any previous standard, with a minimum luma motion compensation block size as small as 4×4 .
- **Quarter-sample-accurate motion compensation:** Most prior standards enable half-sample motion vector accuracy at most. The new design improves up on this by adding quarter-sample motion vector accuracy, as first found in an advanced profile of the MPEG-4 Visual (part 2) stan-

dard, but further reduces the complexity of the interpolation processing compared to the prior design.

- **Motion vectors over picture boundaries:** While motion vectors in MPEG-2 and its predecessors were required to point only to areas within the previously-decoded reference picture, the picture boundary extrapolation technique first found as an optional feature in H.263 is included in H.264/AVC.
- **Multiple reference picture motion compensation:** Predictively coded pictures (called “P” pictures) in MPEG-2 and its predecessors used only one previous picture to predict the values in an incoming picture. The new design extends upon the enhanced reference picture selection technique found in H.263++ to enable efficient coding by allowing an encoder to select, for motion compensation purposes, among a larger number of pictures that have been decoded and stored in the decoder. The same extension of referencing capability is also applied to motion-compensated bi-prediction, which is restricted in MPEG-2 to using two specific pictures only (one of these being the previous intra (I) or P picture in display order and the other being the next I or P picture in display order).
- **Decoupling of referencing order from display order:** In prior standards, there was a strict dependency between the ordering of pictures for motion compensation referencing purposes and the ordering of pictures for display purposes. In H.264/AVC, these restrictions are largely removed, allowing the encoder to choose the ordering of pictures for referencing and display purposes with a high degree of flexibility constrained only by a total memory capacity bound imposed to ensure decoding ability. Removal of the restriction also enables removing the extra delay previously associated with bi-predictive coding.
- **Decoupling of picture representation methods from picture referencing capability:** In prior standards, pictures encoded using some encoding methods (namely bi-predictively-encoded pictures) could not be used as references for prediction of other pictures in the video sequence. By removing this restriction, the new standard provides the encoder more flexibility and, in many cases, an ability to use a picture for referencing that is a closer approximation to the picture being encoded.
- **Weighted prediction:** A new innovation in H.264/AVC allows the motion-compensated prediction signal to be weighted and offset by amounts specified by the encoder. This can dramatically improve coding efficiency for scenes containing fades, and can be used flexibly for other purposes as well.
- **Improved “skipped” and “direct” motion inference:** In prior standards, a “skipped” area of a predictively-coded picture could not motion in the scene content. This had a detrimental effect when coding video containing global motion, so the new H.264/AVC design instead infers motion in “skipped” areas. For bi-predictively coded areas (called B slices), H.264/AVC also includes an enhanced motion inference method known as “direct” motion compensation, which improves further on prior “direct” prediction designs found in H.263+ and MPEG-4 Visual.

- **Directional spatial prediction for intra coding:** A new technique of extrapolating the edges of the previously-decoded parts of the current picture is applied in regions of pictures that are coded as intra (i.e., coded without reference to the content of some other picture). This improves the quality of the prediction signal, and also allows prediction from neighboring areas that were not coded using intra coding (something not enabled when using the transform-domain prediction method found in H.263+ and MPEG-4 Visual).
- **In-the-loop deblocking filtering:** Block-based video coding produces artifacts known as blocking artifacts. These can originate from both the prediction and residual difference coding stages of the decoding process. Application of an adaptive deblocking filter is a well-known method of improving the resulting video quality, and when designed well, this can improve both objective and subjective video quality. Building further on a concept from an optional feature of H.263+, the deblocking filter in the H.264/AVC design is brought within the motion-compensated prediction loop, so that this improvement in quality can be used in inter-picture prediction to improve the ability to predict other pictures as well.

In addition to improved prediction methods, other parts of the design were also enhanced for improved coding efficiency, including the following.

- **Small block-size transform:** All major prior video coding standards used a transform block size of 8×8 , while the new H.264/AVC design is based primarily on a 4×4 transform. This allows the encoder to represent signals in a more locally-adaptive fashion, which reduces artifacts known colloquially as “ringing”. (The smaller block size is also justified partly by the advances in the ability to better predict the content of the video using the techniques noted above, and by the need to provide transform regions with boundaries that correspond to those of the smallest prediction regions.)
- **Hierarchical block transform:** While in most cases, using the small 4×4 transform block size is perceptually beneficial, there are some signals that contain sufficient correlation to call for some method of using a representation with longer basis functions. The H.264/AVC standard enables this in two ways: 1) by using a hierarchical transform to extend the effective block size used for low-frequency chroma information to an 8×8 array and 2) by allowing the encoder to select a special coding type for intra coding, enabling extension of the length of the luma transform for low-frequency information to a 16×16 block size in a manner very similar to that applied to the chroma.
- **Short word-length transform:** All prior standard designs have effectively required encoders and decoders to use more complex processing for transform computation. While previous designs have generally required 32-bit processing, the H.264/AVC design requires only 16-bit arithmetic.
- **Exact-match inverse transform:** In previous video coding standards, the transform used for representing the video was generally specified only within an error tolerance bound, due to the impracticality of obtaining an exact match to the ideal specified inverse transform. As a result, each decoder design would produce slightly different decoded video, causing a “drift” between encoder and decoder representation of the video and reducing effective video quality. Building on a path laid out as an optional feature in the H.263++ effort, H.264/AVC is the first standard to achieve exact equality of decoded video content from all decoders.
- **Arithmetic entropy coding:** An advanced entropy coding method known as arithmetic coding is included in H.264/AVC. While arithmetic coding was previously found as an optional feature of H.263, a more effective use of this technique is found in H.264/AVC to create a very powerful entropy coding method known as CABAC (context-adaptive binary arithmetic coding).
- **Context-adaptive entropy coding:** The two entropy coding methods applied in H.264/AVC, termed CAVLC (context-adaptive variable-length coding) and CABAC, both use context-based adaptivity to improve performance relative to prior standard designs.

Robustness to data errors/losses and flexibility for operation over a variety of network environments is enabled by a number of design aspects new to the H.264/AVC standard, including the following highlighted features.

- **Parameter set structure:** The parameter set design provides for robust and efficient conveyance header information. As the loss of a few key bits of information (such as sequence header or picture header information) could have a severe negative impact on the decoding process when using prior standards, this key information was separated for handling in a more flexible and specialized manner in the H.264/AVC design.
- **NAL unit syntax structure:** Each syntax structure in H.264/AVC is placed into a logical data packet called a NAL unit. Rather than forcing a specific bitstream interface to the system as in prior video coding standards, the NAL unit syntax structure allows greater customization of the method of carrying the video content in a manner appropriate for each specific network.
- **Flexible slice size:** Unlike the rigid slice structure found in MPEG-2 (which reduces coding efficiency by increasing the quantity of header data and decreasing the effectiveness of prediction), slice sizes in H.264/AVC are highly flexible, as was the case earlier in MPEG-1.
- **Flexible macroblock ordering (FMO):** A new ability to partition the picture into regions called slice groups has been developed, with each slice becoming an independently-decodable subset of a slice group. When used effectively, flexible macroblock ordering can significantly enhance robustness to data losses by managing the spatial relationship between the regions that are coded in each slice. (FMO can also be used for a variety of other purposes as well.)
- **Arbitrary slice ordering (ASO):** Since each slice of a coded picture can be (approximately) decoded independently of the other slices of the picture, the H.264/AVC

design enables sending and receiving the slices of the picture in any order relative to each other. This capability, first found in an optional part of H.263+, can improve end-to-end delay in real-time applications, particularly when used on networks having out-of-order delivery behavior (e.g., internet protocol networks).

- **Redundant pictures:** In order to enhance robustness to data loss, the H.264/AVC design contains a new ability to allow an encoder to send redundant representations of regions of pictures, enabling a (typically somewhat degraded) representation of regions of pictures for which the primary representation has been lost during data transmission.
- **Data Partitioning:** Since some coded information for representation of each region (e.g., motion vectors and other prediction information) is more important or more valuable than other information for purposes of representing the video content, H.264/AVC allows the syntax of each slice to be separated into up to three different partitions for transmission, depending on a categorization of syntax elements. This part of the design builds further on a path taken in MPEG-4 Visual and in an optional part of H.263 + +. Here, the design is simplified by having a single syntax with partitioning of that same syntax controlled by a specified categorization of syntax elements.
- **SP/SI synchronization/switching pictures:** The H.264/AVC design includes a new feature consisting of picture types that allow exact synchronization of the decoding process of some decoders with an ongoing video stream produced by other decoders without penalizing all decoders with the loss of efficiency resulting from sending an I picture. This can enable switching a decoder between representations of the video content that used different data rates, recovery from data losses or errors, as well as enabling trick modes such as fast-forward, fast-reverse, etc.

In Sections III and IV, a more detailed description of the key features is given.

III. NAL

The NAL is designed in order to provide “network friendliness” to enable simple and effective customization of the use of the VCL for a broad variety of systems.

The NAL facilitates the ability to map H.264/AVC VCL data to transport layers such as:

- RTP/IP for any kind of real-time wire-line and wireless Internet services (conversational and streaming);
- File formats, e.g., ISO MP4 for storage and MMS;
- H.32X for wireline and wireless conversational services;
- MPEG-2 systems for broadcasting services, etc.

The full degree of customization of the video content to fit the needs of each particular application is outside the scope of the H.264/AVC standardization effort, but the design of the NAL anticipates a variety of such mappings. Some key concepts of the NAL are NAL units, byte stream, and packet format uses of NAL units, parameter sets, and access units. A short description of these concepts is given below whereas a more detailed

description including error resilience aspects is provided in [6] and [7].

A. NAL Units

The coded video data is organized into NAL units, each of which is effectively a packet that contains an integer number of bytes. The first byte of each NAL unit is a header byte that contains an indication of the type of data in the NAL unit, and the remaining bytes contain payload data of the type indicated by the header.

The payload data in the NAL unit is interleaved as necessary with *emulation prevention bytes*, which are bytes inserted with a specific value to prevent a particular pattern of data called a *start code prefix* from being accidentally generated inside the payload.

The NAL unit structure definition specifies a generic format for use in both packet-oriented and bitstream-oriented transport systems, and a series of NAL units generated by an encoder is referred to as a NAL unit stream.

B. NAL Units in Byte-Stream Format Use

Some systems (e.g., H.320 and MPEG-2/H.222.0 systems) require delivery of the entire or partial NAL unit stream as an ordered stream of bytes or bits within which the locations of NAL unit boundaries need to be identifiable from patterns within the coded data itself.

For use in such systems, the H.264/AVC specification defines a byte stream format. In the byte stream format, each NAL unit is prefixed by a specific pattern of three bytes called a start code prefix. The boundaries of the NAL unit can then be identified by searching the coded data for the unique start code prefix pattern. The use of emulation prevention bytes guarantees that start code prefixes are unique identifiers of the start of a new NAL unit.

A small amount of additional data (one byte per video picture) is also added to allow decoders that operate in systems that provide streams of bits without alignment to byte boundaries to recover the necessary alignment from the data in the stream.

Additional data can also be inserted in the byte stream format that allows expansion of the amount of data to be sent and can aid in achieving more rapid byte alignment recovery, if desired.

C. NAL Units in Packet-Transport System Use

In other systems (e.g., internet protocol/RTP systems), the coded data is carried in packets that are framed by the system transport protocol, and identification of the boundaries of NAL units within the packets can be established without use of start code prefix patterns. In such systems, the inclusion of start code prefixes in the data would be a waste of data carrying capacity, so instead the NAL units can be carried in data packets without start code prefixes.

D. VCL and Non-VCL NAL Units

NAL units are classified into VCL and non-VCL NAL units. The VCL NAL units contain the data that represents the values of the samples in the video pictures, and the non-VCL NAL units contain any associated additional information such as parameter sets (important header data that can apply to a large

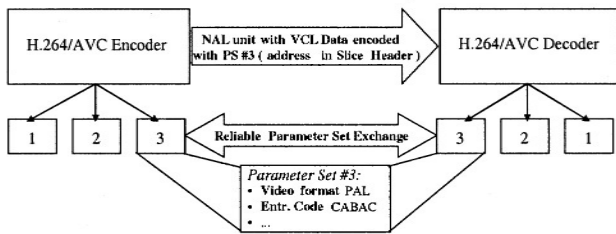


Fig. 3. Parameter set use with reliable "out-of-band" parameter set exchange.

number of VCL NAL units) and supplemental enhancement information (timing information and other supplemental data that may enhance usability of the decoded video signal but are not necessary for decoding the values of the samples in the video pictures).

E. Parameter Sets

A parameter set is supposed to contain information that is expected to rarely change and offers the decoding of a large number of VCL NAL units. There are two types of parameter sets:

- sequence parameter sets, which apply to a series of consecutive coded video pictures called a coded video sequence;
- picture parameter sets, which apply to the decoding of one or more individual pictures within a coded video sequence.

The sequence and picture parameter-set mechanism decouples the transmission of infrequently changing information from the transmission of coded representations of the values of the samples in the video pictures. Each VCL NAL unit contains an identifier that refers to the content of the relevant picture parameter set and each picture parameter set contains an identifier that refers to the content of the relevant sequence parameter set. In this manner, a small amount of data (the identifier) can be used to refer to a larger amount of information (the parameter set) without repeating that information within each VCL NAL unit.

Sequence and picture parameter sets can be sent well ahead of the VCL NAL units that they apply to, and can be repeated to provide robustness against data loss. In some applications, parameter sets may be sent within the channel that carries the VCL NAL units (termed "in-band" transmission). In other applications (see Fig. 3), it can be advantageous to convey the parameter sets "out-of-band" using a more reliable transport mechanism than the video channel itself.

F. Access Units

A set of NAL units in a specified form is referred to as an access unit. The decoding of each access unit results in one decoded picture. The format of an access unit is shown in Fig. 4.

Each access unit contains a set of VCL NAL units that together compose a *primary coded picture*. It may also be prefixed with an *access unit delimiter* to aid in locating the start of the access unit. Some *supplemental enhancement information* containing data such as picture timing information may also precede the primary coded picture.

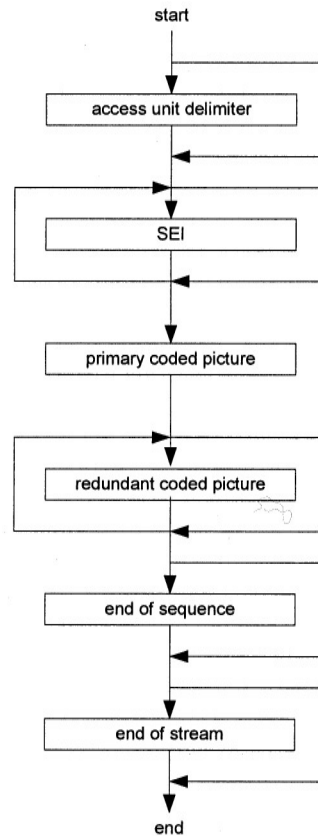


Fig. 4. Structure of an access unit.

The primary coded picture consists of a set of VCL NAL units consisting of *slices* or *slice data partitions* that represent the samples of the video picture.

Following the primary coded picture may be some additional VCL NAL units that contain redundant representations of areas of the same video picture. These are referred to as *redundant coded pictures*, and are available for use by a decoder in recovering from loss or corruption of the data in the primary coded pictures. Decoders are not required to decode redundant coded pictures if they are present.

Finally, if the coded picture is the last picture of a coded video sequence (a sequence of pictures that is independently decodable and uses only one sequence parameter set), an *end of sequence* NAL unit may be present to indicate the end of the sequence; and if the coded picture is the last coded picture in the entire NAL unit stream, an *end of stream* NAL unit may be present to indicate that the stream is ending.

G. Coded Video Sequences

A coded video sequence consists of a series of access units that are sequential in the NAL unit stream and use only one sequence parameter set. Each coded video sequence can be decoded independently of any other coded video sequence, given the necessary parameter set information, which may be conveyed "in-band" or "out-of-band". At the beginning of a coded video sequence is an *instantaneous decoding refresh* (IDR) access unit. An IDR access unit contains an *intra* picture, a coded

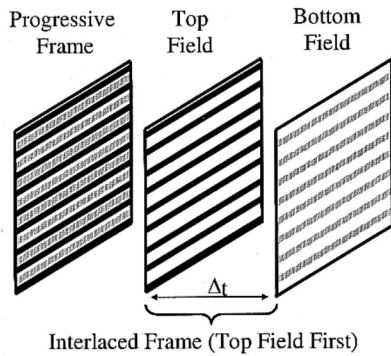


Fig. 5. Progressive and interlaced frames and fields.

picture that can be decoded without decoding any previous pictures in the NAL unit stream, and the presence of an IDR access unit indicates that no subsequent picture in the stream will require reference to pictures prior to the intra picture it contains in order to be decoded.

A NAL unit stream may contain one or more coded video sequences.

IV. VCL

As in all prior ITU-T and ISO/IEC JTC1 video standards since H.261 [3], the VCL design follows the so-called block-based hybrid video coding approach (as depicted in Fig. 8), in which each coded picture is represented in block-shaped units of associated luma and chroma samples called *macroblocks*. The basic source-coding algorithm is a hybrid of inter-picture prediction to exploit temporal statistical dependencies and transform coding of the prediction residual to exploit spatial statistical dependencies. There is no single coding element in the VCL that provides the majority of the significant improvement in compression efficiency in relation to prior video coding standards. It is rather a plurality of smaller improvements that add up to the significant gain.

A. Pictures, Frames, and Fields

A coded video sequence in H.264/AVC consists of a sequence of *coded pictures*. A coded picture in [1] can represent either an entire *frame* or a single *field*, as was also the case for MPEG-2 video.

Generally, a frame of video can be considered to contain two interleaved fields, a top and a bottom field. The top field contains even-numbered rows 0, 2, ..., $H/2-1$ with H being the number of rows of the frame. The bottom field contains the odd-numbered rows (starting with the second line of the frame). If the two fields of a frame were captured at different time instants, the frame is referred to as an interlaced frame, and otherwise it is referred to as a progressive frame (see Fig. 5). The coding representation in H.264/AVC is primarily agnostic with respect to this video characteristic, i.e., the underlying interlaced or progressive timing of the original captured pictures. Instead, its coding specifies a representation based primarily on geometric concepts rather than being based on timing.

B. YCbCr Color Space and 4:2:0 Sampling

The human visual system seems to perceive scene content in terms of brightness and color information separately, and with greater sensitivity to the details of brightness than color. Video transmission systems can be designed to take advantage of this. (This is true of conventional analog TV systems as well as digital ones.) In H.264/AVC as in prior standards, this is done by using a YCbCr color space together with reducing the sampling resolution of the Cb and Cr chroma information.

The video color space used by H.264/AVC separates a color representation into three components called Y, Cb, and Cr. Component Y is called *luma*, and represents brightness. The two *chroma* components Cb and Cr represent the extent to which the color deviates from gray toward blue and red, respectively. (The terms luma and chroma are used in this paper and in the standard rather than the terms luminance and chrominance, in order to avoid the implication of the use of linear light transfer characteristics that is often associated with the terms luminance and chrominance.)

Because the human visual system is more sensitive to luma than chroma, H.264/AVC uses a sampling structure in which the chroma component has one fourth of the number of samples than the luma component (half the number of samples in both the horizontal and vertical dimensions). This is called 4:2:0 sampling with 8 bits of precision per sample. The sampling structure used is the same as in MPEG-2 Main-profile video. (Proposals for extension of the standard to also support higher-resolution chroma and a larger number of bits per sample are currently being considered.)

C. Division of the Picture Into Macroblocks

A picture is partitioned into fixed-size macroblocks that each cover a rectangular picture area of 16×16 samples of the luma component and 8×8 samples of each of the two chroma components. This partitioning into macroblocks has been adopted into all previous ITU-T and ISO/IEC JTC1 video coding standards since H.261 [3]. Macroblocks are the basic building blocks of the standard for which the decoding process is specified. The basic coding algorithm for a macroblock is described after we explain how macroblocks are grouped into slices.

D. Slices and Slice Groups

Slices are a sequence of macroblocks which are processed in the order of a raster scan when not using FMO which is described in the next paragraph. A picture maybe split into one or several slices as shown in Fig. 6. A picture is therefore a collection of one or more slices in H.264/AVC. Slices are self-contained in the sense that given the active sequence and picture parameter sets, their syntax elements can be parsed from the bitstream and the values of the samples in the area of the picture that the slice represents can be correctly decoded without use of data from other slices provided that utilized reference pictures are identical at encoder and decoder. Some information from other slices maybe needed to apply the deblocking filter across slice boundaries.

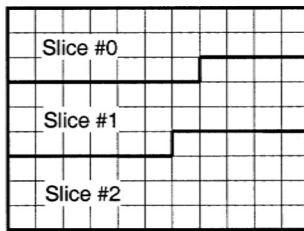


Fig. 6. Subdivision of a picture into slices when not using FMO.

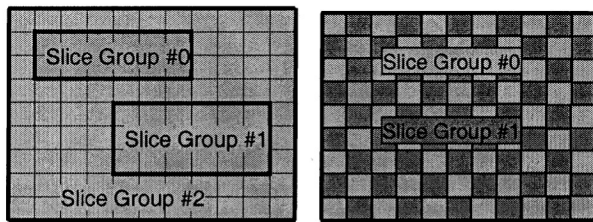


Fig. 7. Subdivision of a QCIF frame into slices when utilizing FMO.

FMO modifies the way how pictures are partitioned into slices and macroblocks by utilizing the concept of *slice groups*. Each slice group is a set of macroblocks defined by a *macroblock to slice group map*, which is specified by the content of the picture parameter set and some information from slice headers. The macroblock to slice group map consists of a slice group identification number for each macroblock in the picture, specifying which slice group the associated macroblock belongs to. Each slice group can be partitioned into one or more slices, such that a slice is a sequence of macroblocks within the same slice group that is processed in the order of a raster scan within the set of macroblocks of a particular slice group. (The case when FMO is not in use can be viewed as the simple special case of FMO in which the whole picture consists of a single slice group.)

Using FMO, a picture can be split into many macroblock scanning patterns such as interleaved slices, a dispersed macroblock allocation, one or more “foreground” slice groups and a “leftover” slice group, or a checker-board type of mapping. The latter two are illustrated in Fig. 7. The left-hand side macroblock to slice group mapping has been demonstrated for use in region-of-interest type of coding applications. The right-hand side macroblock to slice group mapping has been demonstrated useful for concealment in video conferencing applications where slice group #0 and slice group #1 are transmitted in separate packets and one of them is lost. For more details on the use of FMO, see [14].

Regardless of whether FMO is in use or not, each slice can be coded using different coding types as follows.

- **I slice:** A slice in which all macroblocks of the slice are coded using intra prediction.
- **P slice:** In addition to the coding types of the I slice, some macroblocks of the P slice can also be coded using inter prediction with at most *one* motion-compensated prediction signal per prediction block.
- **B slice:** In addition to the coding types available in a P slice, some macroblocks of the B slice can also be coded

using inter prediction with *two* motion-compensated prediction signals per prediction block.

The above three coding types are very similar to those in previous standards with the exception of the use of reference pictures as described below. The following two coding types for slices are new.

- **SP slice:** A so-called switching P slice that is coded such that efficient switching between different pre-coded pictures becomes possible.
- **SI slice:** A so-called switching I slice that allows an exact match of a macroblock in an SP slice for random access and error recovery purposes.

For details on the novel concept of SP and SI slices, the reader is referred to [5], while the other slice types are further described below.

E. Encoding and Decoding Process for Macroblocks

All luma and chroma samples of a macroblock are either spatially or temporally predicted, and the resulting prediction residual is encoded using transform coding. For transform coding purposes, each color component of the prediction residual signal is subdivided into smaller 4×4 blocks. Each block is transformed using an integer transform, and the transform coefficients are quantized and encoded using entropy coding methods.

Fig. 8 shows a block diagram of the VCL for a macroblock. The input video signal is split into macroblocks, the association of macroblocks to slice groups and slices is selected, and then each macroblock of each slice is processed as shown. An efficient parallel processing of macroblocks is possible when there are various slices in the picture.

F. Adaptive Frame/Field Coding Operation

In interlaced frames with regions of moving objects or camera motion, two adjacent rows tend to show a reduced degree of statistical dependency when compared to progressive frames in. In this case, it may be more efficient to compress each field separately. To provide high coding efficiency, the H.264/AVC design allows encoders to make any of the following decisions when coding a frame.

- 1) To combine the two fields together and to code them as one single coded frame (frame mode).
- 2) To not combine the two fields and to code them as separate coded fields (field mode).
- 3) To combine the two fields together and compress them as a single frame, but when coding the frame to split the pairs of two vertically adjacent macroblocks into either pairs of two field or frame macroblocks before coding them.

The choice between the three options can be made adaptively for each frame in a sequence. The choice between the first two options is referred to as picture-adaptive frame/field (PAFF) coding. When a frame is coded as two fields, each field is partitioned into macroblocks and is coded in a manner very similar to a frame, with the following main exceptions:

- motion compensation utilizes reference fields rather than reference frames;
- the zig-zag scan of transform coefficients is different;

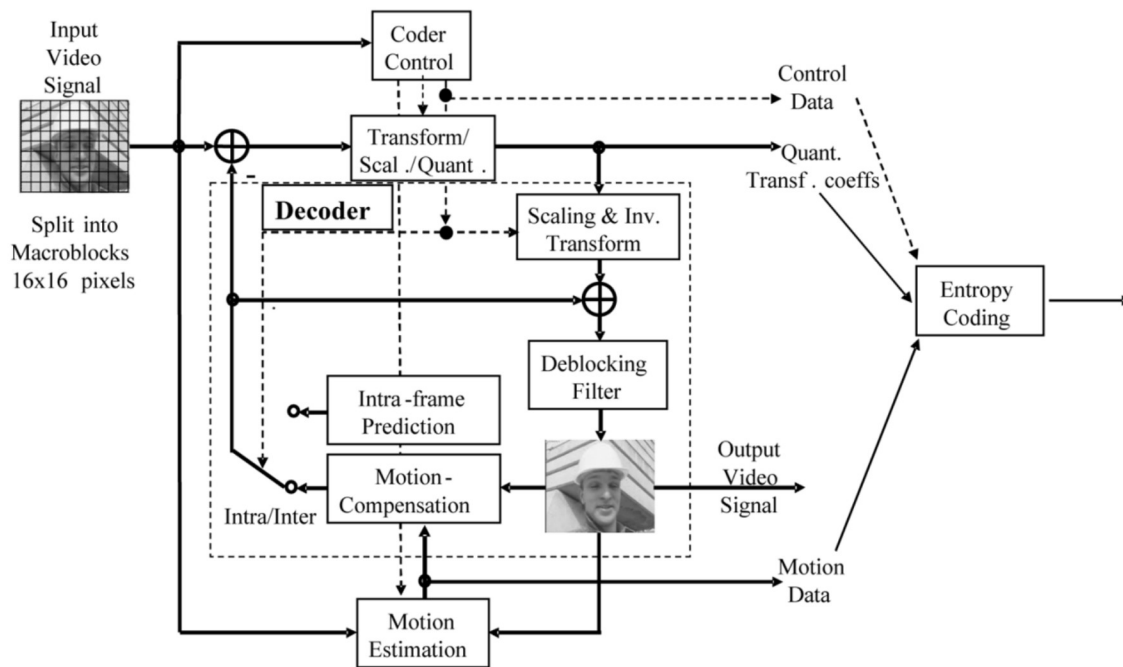


Fig. 8. Basic coding structure for H.264/AVC for a macroblock.

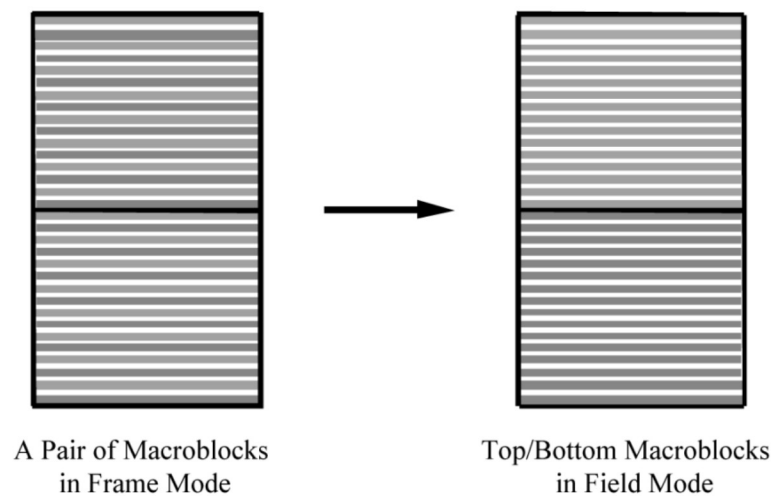


Fig. 9. Conversion of a frame macroblock pair into a field macroblock pair.

- the strong deblocking strength is not used for filtering horizontal edges of macroblocks in fields, because the field rows are spatially twice as far apart as frame rows and the length of the filter thus covers a larger spatial area.

During the development of the H.264/AVC standard, PAFF coding was reported to reduce bit rates in the range of 16% to 20% over frame-only coding mode for ITU-R 601 resolution sequences like “Canoa”, “Rugby”, etc.

If a frame consists of mixed regions where some regions are moving and others are not, it is typically more efficient to code the nonmoving regions in frame mode and the moving regions in the field mode. Therefore, the frame/field encoding decision can also be made independently for each vertical pair of macroblocks (a 16×32 luma region) in a frame. This coding option is referred to as macroblock-adaptive frame/field (MBAFF)

coding. For a macroblock pair that is coded in frame mode, each macroblock contains frame lines. For a macroblock pair that is coded in field mode, the top macroblock contains top field lines and the bottom macroblock contains bottom field lines. Fig. 9 illustrates the MBAFF macroblock pair concept.

Note that, unlike in MPEG-2, the frame/field decision is made at the macroblock pair level rather than within the macroblock level. The reasons for this choice are to keep the basic macroblock processing structure intact, and to permit motion compensation areas as large as the size of a macroblock.

Each macroblock of a field macroblock pair is processed very similarly to a macroblock within a field in PAFF coding. However, since a mixture of field and frame macroblock pairs may occur within an MBAFF frame, the methods that are used for zig-zag scanning, prediction of motion vectors, prediction

of intra prediction modes, intra-frame sample prediction, deblocking filtering, and context modeling in entropy coding are modified to account for this mixture. The main idea is to preserve as much spatial consistency as possible. It should be noted that the specification of spatial neighbors in MBAFF frames is rather complicated (please refer to [1]) and that in Sections IV-G-L spatial neighbors are only described for non-MBAFF frames.

Another important distinction between MBAFF and PAFF is that in MBAFF, one field cannot use the macroblocks in the other field of the same frame as a reference for motion prediction (because some regions of each field are not yet available when a field macroblock of the other field is coded). Thus, sometimes PAFF coding can be more efficient than MBAFF coding (particularly in the case of rapid global motion, scene change, or intra picture refresh), although the reverse is usually true.

During the development of the standard, MBAFF was reported to reduce bit rates in the range of 14 to 16% over PAFF for ITU-R 601 resolution sequences like "Mobile and Calendar" and "MPEG-4 World News".

G. Intra-Frame Prediction

Each macroblock can be transmitted in one of several coding types depending on the slice-coding type. In all slice-coding types, the following types of intra coding are supported, which are denoted as Intra_4×4 or Intra_16×16 together with chroma prediction and I_PCM prediction modes.

The Intra_4×4 mode is based on predicting each 4×4 luma block separately and is well suited for coding of parts of a picture with significant detail. The Intra_16×16 mode, on the other hand, performs prediction of the whole 16×16 luma block and is more suited for coding very smooth areas of a picture. In addition to these two types of luma prediction, a separate chroma prediction is conducted. As an alternative to Intra_4×4 and Intra_16×16, the I_PCM coding type allows the encoder to simply bypass the prediction and transform coding processes and instead directly send the values of the encoded samples. The I_PCM mode serves the following purposes.

- 1) It allows the encoder to precisely represent the values of the samples.
- 2) It provides a way to accurately represent the values of anomalous picture content without significant data expansion
- 3) It enables placing a hard limit on the number of bits a decoder must handle for a macroblock without harm to coding efficiency

In contrast to some previous video coding standards (namely H.263+ and MPEG-4 Visual), where intra prediction has been conducted in the transform domain intra prediction in H.264/AVC is always conducted in the spatial domain, by referring to neighboring samples of previously-coded blocks which are to the left and/or above the block to be predicted. This may incur error propagation in environments with transmission errors that propagate due to motion compensation into inter-coded macroblocks. Therefore, a constrained intra coding mode can be signaled that allows prediction only from intra-coded neighboring macroblocks.

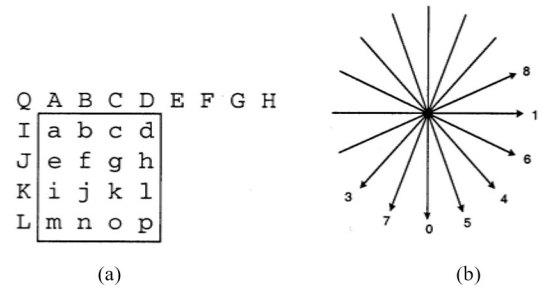


Fig. 10. (a) Intra_4×4 prediction is conducted for samples a-p of a block using samples A-Q. (b) Eight "prediction directions" for Intra_4×4 prediction.

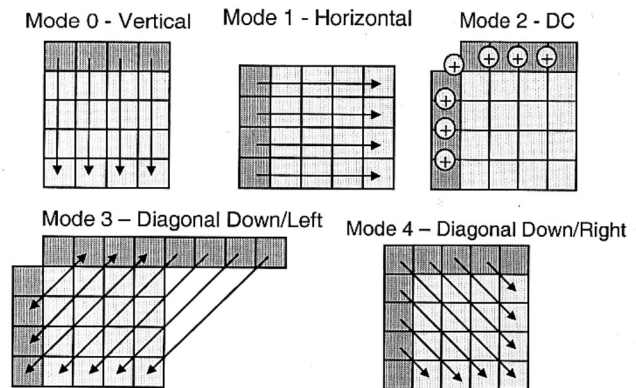


Fig. 11. Five of the nine Intra_4×4 prediction modes.

When using the Intra_4×4 mode, each 4×4 block is predicted from spatially neighboring samples as illustrated on the left-hand side of Fig. 10. The 16 samples of the 4×4 block which are labeled as a-p are predicted using prior decoded samples in adjacent blocks labeled as A-Q. For each 4×4 block, one of nine prediction modes can be utilized. In addition to "DC" prediction (where one value is used to predict the entire 4×4 block), eight directional prediction modes are specified as illustrated on the right-hand side of Fig. 10. Those modes are suitable to predict directional structures in a picture such as edges at various angles.

Fig. 11 shows five of the nine Intra_4×4 prediction modes. For mode 0 (vertical prediction), the samples above the 4×4 block are copied into the block as indicated by the arrows. Mode 1 (horizontal prediction) operates in a manner similar to vertical prediction except that the samples to the left of the 4×4 block are copied. For mode 2 (DC prediction), the adjacent samples are averaged as indicated in Fig. 11. The remaining six modes are diagonal prediction modes which are called diagonal-down-left, diagonal-down-right, vertical-right, horizontal-down, vertical-left, and horizontal-up prediction. As their names indicate, they are suited to predict textures with structures in the specified direction. The first two diagonal prediction modes are also illustrated in Fig. 11. When samples E-H (Fig. 10) that are used for the diagonal-down-left prediction mode are not available (because they have not yet been decoded or they are outside of the slice or not in an intra-coded macroblock in the constrained intra-mode), these samples are replaced by sample D. Note that in earlier draft versions of the Intra_4×4 prediction mode the four samples below sample L were also used for some prediction modes. However, due to the need to reduce memory access