

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMAZON.COM, INC. and AMAZON.COM SERVICES LLC,

Petitioner

v.

NOKIA TECHNOLOGIES OY,

Patent Owner.

IPR2024-01175

U.S. Patent No. 6,856,701

DECLARATION OF ELI SABER, PH.D.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further, that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Date: September 16, 2024

By:



Eli Saber, Ph.D.

TABLE OF CONTENTS

I.	BACKGROUND AND QUALIFICATIONS.....	2
II.	LEVEL OF ORDINARY SKILL.....	7
III.	MATERIALS CONSIDERED	8
IV.	LEGAL STANDARDS.....	9
	A. Priority Date	9
	B. Obviousness.....	10
V.	OVERVIEW OF RELEVANT VIDEO CODING TECHNOLOGIES.....	12
	A. Discrete Cosine Transform (DCT)	14
	B. Quantization	15
	C. Run-Level (Run-Length) Coding	16
	D. Entropy Coding	18
VI.	The '701 Patent	21
	A. Applicant-Admitted Prior Art (AAPA).....	21
	B. Alleged Problem in the Prior Art and Purported Solution	28
	C. The Challenged Claims	29
VII.	CLAIM CONSTRUCTION.....	29
VIII.	DETAILED EXPLANATION OF GROUNDS	30
	A. Ground 1A: Obviousness over Tsai and VCEG-L28 for Claims 1-14, 17-31, and 34-40.....	30
	1. Tsai.....	30
	2. VCEG-L28	39
	3. Claim 1	42
	a. 1[pre].....	42
	b. 1[a]	48
	c. 1[b].....	50
	4. Claim 2	58
	5. Claim 3	59
	6. Claim 4	60

7.	Claim 5	60
8.	Claim 6	61
9.	Claim 7	62
10.	Claim 8	63
11.	Claim 9	65
12.	Claim 10	66
	a. 10[pre]	66
	b. 10[a]	67
	c. 10[b]	67
13.	Claim 11	70
14.	Claim 12	70
15.	Claim 13	70
16.	Claim 14	71
17.	Claim 17	72
18.	Claim 18	72
	a. 18[pre]	72
	b. 18[a]	73
	c. 18[b]	74
	d. 18[c]	75
	e. 18[d]	77
	f. 18[e]	78
19.	Claim 19	79
20.	Claim 20	79
21.	Claim 21	79
22.	Claim 22	80
23.	Claim 23	80
24.	Claim 24	80
25.	Claim 25	80
26.	Claim 26	81

27.	Claim 27	81
a.	27[pre]	81
b.	27[a]	81
c.	27[b]	81
d.	27[c]	82
e.	27[d]	82
f.	27[e]	82
28.	Claim 28	82
29.	Claim 29	83
30.	Claim 30	83
31.	Claim 31	83
32.	Claim 34	83
33.	Claim 35	84
a.	35[pre]	84
b.	35[a]	84
c.	35[b]	84
d.	35[c]	84
e.	35[d]	85
f.	35[e]	85
34.	Claim 36	85
a.	36[pre]	85
b.	36[a]	85
c.	36[b]	86
d.	36[c]	86
e.	36[d]	86
f.	36[e]	86
35.	Claim 37	87
a.	37[pre]	87
b.	37[a]	88

c.	37[b].....	88
36.	Claim 38.....	88
a.	38[pre].....	88
b.	38[a].....	88
c.	38[b].....	89
37.	Claim 39.....	89
a.	39[pre].....	89
b.	39[a].....	89
c.	39[b].....	89
38.	Claim 40.....	90
a.	40[pre].....	90
b.	40[a].....	90
c.	40[b].....	90
B.	Ground 1B: Obviousness over Tsai, VCEG-L28, and VCEG-L13 for Claims 15-16 and 32-33.....	91
1.	VCEG-L13	91
2.	Claim 15	91
3.	Claim 16.....	94
4.	Claim 32.....	95
5.	Claim 33	95
C.	Ground 2: Obviousness over AAPA and Savatier for Claims 1-9, 18-26, 35, 37, and 39.....	95
1.	AAPA.....	95
2.	Savatier.....	95
3.	Claim 1	99
a.	1[pre].....	99
b.	1[a].....	103
c.	1[b].....	105
4.	Claim 2	115
5.	Claim 3	115

6.	Claim 4.....	116
7.	Claim 5.....	116
8.	Claim 6.....	117
9.	Claim 7.....	118
10.	Claim 8.....	119
11.	Claim 9.....	120
12.	Claim 18.....	121
	a. 18[pre].....	121
	b. 18[a].....	121
	c. 18[b].....	122
	d. 18[c].....	123
	e. 18[d].....	124
	f. 18[e].....	124
13.	Claim 19.....	127
14.	Claim 20.....	127
15.	Claim 21.....	127
16.	Claim 22.....	128
17.	Claim 23.....	128
18.	Claim 24.....	128
19.	Claim 25.....	128
20.	Claim 26.....	129
21.	Claim 35.....	129
	a. 35[pre].....	129
	b. 35[a].....	129
	c. 35[b].....	129
	d. 35[c].....	130
	e. 35[d].....	130
	f. 35[e].....	130
22.	Claim 37.....	130

a.	37[pre].....	130
b.	37[a].....	131
c.	37[b].....	131
23.	Claim 39	132
a.	39[pre].....	132
b.	39[a].....	133
c.	39[b].....	133
D.	Ground 3: Obviousness over AAPA and VCEG-L13 for Claims 10-17, 27-34, 36, 38, and 40.	133
1.	Claim 10	133
a.	10[pre].....	133
b.	10[a].....	134
c.	10[b].....	134
2.	Claim 11	136
3.	Claim 12	136
4.	Claim 13	137
5.	Claim 14	137
6.	Claim 15	137
7.	Claim 16	138
8.	Claim 17	138
9.	Claim 27	139
a.	27[pre].....	139
b.	27[a].....	139
c.	27[b].....	139
d.	27[c]	139
e.	27[d].....	140
f.	27[e].....	140
10.	Claim 28	140
11.	Claim 29	140

12.	Claim 30	141
13.	Claim 31	141
14.	Claim 32	141
15.	Claim 33	141
16.	Claim 34	142
17.	Claim 36	142
	g. 36[pre]	142
	h. 36[a]	142
	i. 36[b]	142
	j. 36[c]	143
	k. 36[d]	143
	a. 36[e]	143
18.	Claim 38	143
	a. 38[pre]	143
	b. 38[a]	144
	c. 38[b]	144
19.	Claim 40	144
	a. 40[pre]	144
	b. 40[a]	145
	c. 40[b]	145
IX. CONCLUSION		145
APPENDIX A – MATERIALS CONSIDERED		146
APPENDIX B – LISTING OF CHALLENGED CLAIMS		148

I, Eli Saber, hereby declare as follows:

1. I have been retained on behalf of Amazon.com, Inc. and Amazon.com Services LLC (“Amazon” or “Petitioner”) to offer technical opinions related to U.S. Patent No. 6,856,701 (“the ’701 patent”) (EX1001). I understand that Amazon is requesting that the Patent Trial and Appeal Board (“PTAB” or “Board”) institute an *inter partes* review (“IPR”) proceeding for the ’701 patent. Specifically, I have been asked to consider the validity of claims 1-40 of the ’701 patent (the “Challenged Claims”) in view of prior art and obviousness considerations from the perspective of a person of ordinary skill in the art (“POSITA”) at the time of the invention as it relates to the ’701 patent.

2. I am being compensated for my time at my standard consulting rate of \$500/hour (\$300/hour for travel time). I am also being reimbursed for expenses that I incur during this work. My compensation is not contingent upon the results of my study, the substance of my opinions, or the outcome of any proceeding involving the Challenged Claims. I have no financial interest in the outcome of this matter or in the pending litigation between Petitioner and Nokia Technologies Oy (“Patent Owner” or “PO”).

3. My analysis here is based on my years of education, research, and experience, as well as my investigation and study of relevant materials, including those cited herein.

4. I may rely upon these materials, my knowledge and experience, and/or additional materials to rebut arguments raised by the Patent Owner. Further, I may consider additional documents and information in forming any necessary opinions, including documents that may not yet have been provided.

5. My analysis of the materials in this proceeding is ongoing, and I will continue to review any new material as it is provided. This Declaration represents only those opinions I have formed to date. I reserve the right to revise, supplement, and/or amend my opinions herein based on new information and my continuing analysis of the materials already provided.

I. BACKGROUND AND QUALIFICATIONS

6. I am a Professor in the Department of Electrical and Microelectronic Engineering (EME) and the Chester F. Carlson Center for Imaging Science at the Rochester Institute of Technology (RIT) in Rochester, New York. I also serve as the Director of the Image, Video and Computer Vision Laboratory. I joined RIT's full-time faculty in 2004. I have over thirty-four (34) years of industry and academic experience, thirty (30) of which are in the fields of Computer Vision, Digital Image/Video Processing, Pattern Recognition, and Machine Learning.

7. I received a Ph.D. in Electrical Engineering from the University of Rochester in 1996. My concentration was in Signal/Image/Video Processing, Pattern Recognition, and Computer Vision. I received a Master of Science Degree in

Electrical Engineering from the University of Rochester in 1992 in the same concentration as listed above, and I received a Bachelor of Science Degree in Electrical and Computer Engineering, Summa Cum Laude, from the State University of New York at Buffalo in 1988.

8. Before becoming a full-time professor, I worked for Xerox Corporation for 16 years, from 1988 to 2004. During my years at Xerox, I was responsible for delivering color management, image processing innovations, architectures, and algorithms; xerographic sub-systems for a variety of color products; and control systems for toner production facilities. One of several roles I held at Xerox was Advanced Development Scientist and Manager. In that capacity, I established the Advanced Design Laboratory—an Imaging/Xerographics lab—and provided technical and managerial leadership for the Electrical, Imaging, and Xerographics Department. In another role, as Product Development Scientist and Manager, I led the research and development of image quality metrics for various product platforms. I also led the Image Science, Analysis, and Evaluation area, with 12 to 15 direct reports and a budget of approximately \$2 million.

9. From 1997 until 2004, I was an adjunct faculty member at the Electrical Engineering Department of RIT and at the Electrical & Computer Engineering Department of the University of Rochester. I was responsible for teaching undergraduate and graduate coursework such as Digital Signal Processing, Digital

Image Processing, Digital Video Processing, Pattern Recognition, Detection and Estimation Theory and Communication Systems. I also performed research in multimedia applications, pattern recognition, image understanding and color engineering.

10. Since joining RIT full time, I have been responsible for teaching several undergraduate and graduate courses in Digital Signal Processing, Digital Image Processing, Digital Video Processing, Pattern Recognition, Engineering Analysis, Random Signal & Noise, Advanced Engineering Mathematics, Matrix Methods, Communications, Modern Control Theory, and Linear Systems.

11. I proposed, developed, and introduced the Digital Video Processing and the Advanced Neural Network and Deep Learning courses, and I reestablished the Pattern Recognition course at the Electrical Engineering Department at RIT. These graduate courses were instrumental in providing MS/PhD students with the proper foundation and knowledge in all aspects of the subject areas in order to perform in-depth research in the field and continue to advance the state of the art.

12. The Digital Video Processing course is designed to provide a thorough and comprehensive coverage of the field. Topics covered include the mathematical representation of digital video including motion and time-varying image formation models, spatio-temporal sampling/reconstruction, sampling structure conversions, motion estimation using optical flow, block-based, pel-recursive, Bayesian and point

correspondence type methodologies, segmentation, motion tracking, filtering, restoration, super-resolution, and image/video compression fundamental techniques and algorithms. The course ends with a thorough overview of image/video compression standards including JPEG, H.261, MPEG1, MPEG2 and H.264/AVC (advanced video coding).

13. The Pattern Recognition course, on the other hand, covers pertinent concepts in Bayes decision theory, linear and nonlinear classifiers, artificial neural networks, feature selection/generation/extraction and various types of clustering algorithms and techniques.

14. The Advanced Neural Networks and Deep Learning: Theory, Mathematics and Algorithms is intended to provide an in-depth comprehensive study of the theory, mathematics and algorithms for advanced neural networks and their usage in deep learning. It is primarily targeted for upper graduate students that are planning to pursue advanced studies/research in the field. The course begins with an in-depth study of the mathematical methodologies for neural networks and optimization (zero-, first- and second-order) techniques. It then progresses to provide a thorough study of shallow and deep neural networks architectures and algorithms, activation functions, regression, two- and multi-class classification, unsupervised learning, feature engineering/selection, backpropagation along with applications to a variety of engineering problems.

15. I am currently working to develop a follow-on course (part 2) to the Advanced Neural Networks and Deep Learning course described above. This course is intended to cover convolutional and recurrent style neural networks, principles of feature engineering, kernel methods, tree-based learners/random forest classifiers, graph neural networks, and generative adversarial networks. This course is being targeted for the first time in Fall 2025.

16. My current research focuses on machine learning for object tracking/remote sensing applications; image/video processing for multimedia, military & biomedical applications; and computer vision and three-dimensional scene reconstruction. As a principal investigator (PI) or co-principal investigator, I have acquired almost \$6 million in research funding since joining RIT and have managed multiple government grants from the Department of Defense as well as several corporate grants from Hewlett-Packard, Lenel, and Data Physics. I am currently managing (as PI) a multiyear government grant from the Department of Defense titled “Target Detection/Tracking and Activity Recognition from Multimodal Data.” In this project, we propose to develop operational target detection/tracking techniques and activity recognition/understanding methods that can leverage multimodal data in a fusion framework using deep learning algorithms while providing accurate and near real-time performance.

17. In 2012, I was awarded the Trustees Scholarship, the highest award at

RIT with regard to research recognition.

18. I am the author or co-author of thirty-eight (38) peer-reviewed journal publications.

19. I have also authored or co-authored 102 conference and workshop publications and a book entitled Advanced Linear Algebra for Engineers with MATLAB, published by CRC Press in February 2009, and am a named inventor on multiple U.S. and foreign patents.

20. I am a senior member of the Institute of Electrical and Electronic Engineers and a member of the IEEE Signal Processing Society, the Electrical Engineering Honor Society, Eta Kappa Nu.

21. A detailed record of my professional qualifications, including a list of publications, awards, and professional activities, is set forth in my curriculum vitae submitted with this Declaration as EX1004.

II. LEVEL OF ORDINARY SKILL

22. In rendering the opinions set forth in this Declaration, I was asked to consider the patent claims and the prior art through the eyes of a POSITA at the time of the alleged invention, which I understand to be September 14, 2001. I understand that the factors considered in determining the ordinary level of skill in a field of art include the education and experience of persons working in the field, the types of problems encountered in the field, the teachings of the prior art, and the

sophistication of the technology at the time of the alleged invention. I understand that a POSITA is not a specific, real individual, but rather is a hypothetical individual having the qualities reflecting the factors above. I understand that a POSITA would also have knowledge from the teachings of the prior art, including the art cited below.

23. Taking these factors into consideration, it is my opinion that, at the relevant time, a POSITA relating to the technology of the '701 patent would have had a (1) Bachelor's degree in Electrical Engineering, Computer Engineering, Computer Science, or a comparable field of study such as Physics, and (2) approximately two to three years of practical experience with video and/or image processing or coding. In my opinion, additional experience could substitute for the level of education, and vice-versa.

24. Before September 14, 2001, my level of skill in the art was at least that of a POSITA. I am qualified to provide opinions concerning what a POSITA would have known and understood at that time, and my analysis and conclusions herein are from the perspective of a POSITA as of that date.

III. MATERIALS CONSIDERED

25. In forming my opinions expressed herein, I have relied upon my education and experience, and I have considered the materials cited herein, including those itemized in the table of Materials Considered attached as **Appendix A** to this

Declaration.

IV. LEGAL STANDARDS

26. I am not a lawyer and do not provide any legal opinions, but I have been advised that certain legal standards are to be applied by technical experts in forming opinions regarding the meaning and validity of patent claims. I have applied the legal standards described below.

27. I understand that assessing a U.S. patent's validity based on a prior art analysis involves two steps. First, one must assess what meaning a POSITA would have given the terms. Second, after assessing the meaning of the claim terms, one may then assess validity by comparing a patent claim to the "prior art." I understand that the teaching of the prior art is viewed through the eyes of a POSITA at the time the invention was made. My analysis as to what constitutes a relevant POSITA is set forth above.

A. Priority Date

28. I understand that the "priority date" (or "earliest effective filing date") of a patent is the date on which it is filed or the date on which an earlier application was filed if the patentee properly claims the benefit of the earlier application's filing date. For my analysis in this Declaration, I have been asked to assume that the '701 patent is entitled to a priority date of September 14, 2001.

B. Obviousness

29. My understanding is that a patent claim is invalid as obvious only if the subject matter of the claimed invention “as a whole” would have been obvious to a POSITA at the time the invention was made. To determine the differences between a prior art reference (or a proposed combination of prior art references) and the claims, the question of obviousness is not whether the differences themselves would have been obvious but whether the claimed invention as a whole would have been obvious. Also, obviousness grounds cannot be sustained by mere conclusory statements. Rather, it is necessary to provide some articulated reasoning with rational underpinning to support the legal conclusion of obviousness.

30. I understand that a patent claim that comprises several elements is not proved obvious by simply showing that each of its elements was independently known in the prior art. In my evaluation of whether any claim of the '701 patent would have been obvious, I considered whether a POSITA would have had a reason with a rational basis that would have motivated that POSITA to combine the elements or concepts from the prior art in the same way as in the claimed invention.

31. I understand that there is no single way to define the line between true inventiveness on the one hand—which is patentable—and the application of common sense and ordinary skill to solve a problem on the other hand—which is not patentable. For instance, factors such as market forces or other design incentives

may be the source of what produced a change rather than true inventiveness.

32. I understand that the decision-maker may consider whether the change was merely the predictable result of using prior art elements according to their known functions or whether it was the result of true inventiveness. And the decision-maker may also consider whether there is some teaching or suggestion in the prior art to make the modification or combination of elements recited in the claim at issue. Also, the decision-maker may consider whether the innovation applies a known technique that had been used to improve a similar device or method. The decision-maker may also consider whether the claimed invention would have been obvious to try, meaning that the claimed innovation was one of a relatively small number of possible approaches to the problem with a reasonable expectation of success by those skilled in the art.

33. I understand that in considering obviousness, it is important to be careful not to engage in hindsight and that many true inventions might seem obvious after the fact. I understand that obviousness is considered from the perspective of a POSITA at the time the claimed invention was made and not based on what is known today or what is learned from the teaching of the patent.

34. I understand that the following factors must be evaluated to determine whether a claim is obvious: the scope and content of the prior art; the difference or differences, if any, between the claim of the patent and the prior art; the level of

ordinary skill in the art at the time the claimed invention was made; and, if available, the objective indicia of non-obviousness, also known as “secondary considerations.”

35. I understand that the secondary considerations can include: commercial success of a product due to the merits of the claimed invention, a long-felt need for the solution provided by the claimed invention, unsuccessful attempts by others to find the solution provided by the claimed invention; copying of the claimed invention by others, unexpected and superior results from the claimed invention; acceptance by others of the claimed invention as shown by praise from others in the field or from the licensing of the claimed invention, teaching away from the conventional wisdom in the art at the time of the invention, independent invention of the claimed invention by others before or at about the same time as the named inventor thought of it, and other evidence tending to show obviousness.

36. I understand that, to establish a secondary consideration, the evidence must demonstrate a nexus between that secondary consideration and the claimed invention.

V. OVERVIEW OF RELEVANT VIDEO CODING TECHNOLOGIES

37. Digital video, like analog video, comprises a sequence of still images (frames) that the human vision system perceives to be motion. Each frame is divided into a matrix of points called pixels (short for picture elements) representing color (chrominance) and brightness (luminance). Riley, 16-18.

38. Image and video coding algorithms aim to compress input data as much as possible while preserving a sufficient level of perceptual quality in the displayed images or video. LeGall, 47-49; Tsai, 1:28-33. Fortunately, digital video in its raw format contains a lot of redundant information. LeGall, 51-54; Riley, 19.

39. Redundant information found in uncompressed digital video can be temporally or spatially redundant. Temporal redundancy occurs when, within a given sequence of frames (scene) being imaged by a video system, motion occurs only a fraction of the time and typically over only a portion of the frames spatially. Thus, the pixel values will be, in general, almost identical from one frame to the next over most of the frames. For example, sportscasters sitting at a desk demonstrate little motion over only a portion of the scene. Spatial redundancy occurs when, within a given frame being imaged by a video system, neighboring pixels display similar visual information, such as color or intensity. Thus, the pixel values will be almost identical across adjacent areas within the frame. For example, if a video frame contains a sky section, or a uniformly painted wall, adjacent pixels within the frame will have almost identical color.

40. Video coding techniques exploit both spatial and temporal redundancy, reducing the number of bits needed for storage or transmission while maintaining desired video quality. Riley, 19. For example, in the widely used video coding standards developed by the Moving Picture Experts Group (“MPEG”), pixels are

grouped into blocks, which are then processed using a combination of motion-compensated prediction (to reduce temporal redundancy) and transform coding of the remaining spatial information (to reduce spatial redundancy). LeGall, 51; Tsai, 1:48-2:22. Video coding standards developed by the International Telecommunication Union (“ITU”) for use in videoconferencing and videotelephony applications (e.g., H.261, H.263, and H.26L) also use a combination of motion-compensated prediction and transform coding of the remaining spatial information. LeGall, 48, 52, 54-55; Riley, 34-38; EX1001, 11:36-51, FIGS. 1-2.

41. The ’701 patent is directed to transform coding techniques. *See, e.g.*, EX1001, 17:8-10 (“In principle the invention can be applied in any image coding system in which block-based transform coding and context-based arithmetic coding are used.”). In a transform coding type system, pixels are grouped into blocks that are converted into another domain to produce a set of transform coefficients, which are then coded and transmitted. An effective transform will compact the energy in the block of pixels (i.e., the “useful information” in the block will be concentrated into a few of the coefficients). Riley, 21. Compression is achieved by quantizing the coefficients so that the “useful” coefficients are transmitted and the remaining coefficients are discarded. Riley, 21.

A. Discrete Cosine Transform (DCT)

42. By the ’701 patent’s earliest priority date, transform coding was a

common method for compressing image information, with the discrete cosine transform (“DCT”) being extensively used for image and video compression systems. Riley, 21; LeGall, 51, 54-55. The DCT provides energy compaction that is close to the optimum for many types of images. *See* Riley, 21; LeGall, 54 (“The DCT is the best of the orthogonal transforms with a fast algorithm, and a very close approximation to the optimal for a large class of images.”). By the ’701 patent’s earliest priority date, several fast algorithms were known for calculating the DCT of a block of pixels. Riley, 21, 23-24.

43. The DCT divides an image into regular-sized pixel blocks, e.g., 8x8 pixels. Riley, 22. Each block of pixels is converted into a block of transform coefficients (of the same dimensions) that represents the block in the spatial frequency domain. Riley, 22-24; LeGall, 54-55.

44. Although the DCT changes the representation of the image data, it does not inherently compress it until further steps like quantization are applied. Riley, 23; LeGall, 54, Figure 4.

B. Quantization

45. The DCT coefficients are quantized so that the near-zero coefficients are set to zero and the remaining coefficients are represented with a reduced precision. Riley, 24; Savatier, 3:55-60.

46. Quantization is applied to each DCT coefficient by dividing it by a

scaling factor and rounding the result to the nearest integer, resulting in loss of information, but also in compression since most of the values in each block become zero. Riley, 24. Quantization is inherently lossy; it simplifies high-frequency components more aggressively since the human visual system is less sensitive to detail loss in higher frequencies. LeGall, 55. Coarser quantization leads to higher compression, but lower image quality. Riley, 24.

47. After they are quantized, the DCT coefficients are reordered using a zigzag scanning pattern that orders them beginning with the lower-frequency (more significant) coefficients and continuing through the higher-frequency (less significant) ones. LeGall, 54, Figure 4; Riley, 31-33. The zigzag pattern tends to group non-zero coefficients toward the beginning of the scan, followed by runs of zeros at the end. This organization helps improve compression efficiency, as it leads to longer runs of zero coefficients, which are efficiently coded using run-level coding. LeGall, 54-55; Riley, 33.

C. Run-Level (Run-Length) Coding¹

48. To reduce the overhead of signaling zero coefficients, the sequence of quantized DCT coefficients is represented using pairs of value referred to as “level” and “run,” where level is the value of a non-zero coefficient and run is the number

¹ Run/level coding is sometimes called “run-length” coding. *See, e.g.,* LeGall, 54.

of consecutive zero coefficients preceding the non-zero coefficient. Tsai, 2:40-45; Riley, 24.

49. The Tsai reference provides an overview of the run-level/length coding process with reference to an exemplary sequence of quantized DCT coefficients shown in FIG. 4B (below). Tsai, 9:49-10:9.

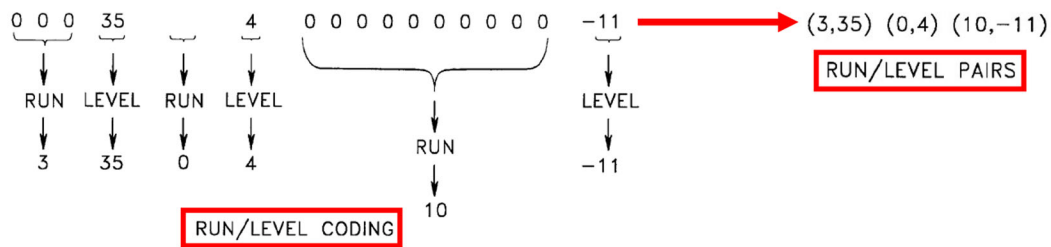
0 0 0 35 4 0 0 0 0 0 0 0 0 0 0 -11

SEQUENTIAL INTEGER LIST

Tsai, FIG. 4B (simplified).

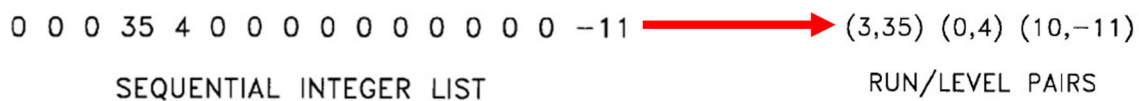
50. The run-level/length coding process is illustrated in FIGS. 4C-4D of Tsai (below). First, the run-length coder counts the leading zeros. Because there are three leading zeros before encountering a nonzero value, the first run value has a value of three (3). The first non-zero value encountered is 35; thus, the first level value has a value of 35. The first run-level pair is thus **(3, 35)**. Next, the run-length coder scans again for zeros. This time, however, there are no zeros before encountering a non-zero value. Thus, the next run value is zero. Because four (4) is the next non-zero value encountered, the next level value is four (4). The second run-level pair is thus **(0, 4)**. The run-length coder again scans for zeros, finding ten zeros before encountering a non-zero value. Thus, the next run value is 10. The non-zero value encountered immediately after the ten zeros is -11; therefore the next level

value is -11. The third run-level pair is thus **(10, -11)**.



Tsai, FIGS. 4C and 4D (simplified and annotated).

51. FIG. 4D illustrates the three run/level pairs created by the run-length coder from the sequential integer list in FIG. 4B:



Tsai, FIGS. 4B and 4D (simplified and annotated).

52. Next, entropy coding is applied to the run-level pairs to further reduce the amount of coded data.

D. Entropy Coding

53. Entropy coding reduces the number of bits needed to represent data by assigning variable-length codes based on symbol² frequencies. More frequent symbols are encoded with shorter codes, while less frequent symbols are given longer ones. LeGall, 55. Huffman coding is a well-known example of a variable-

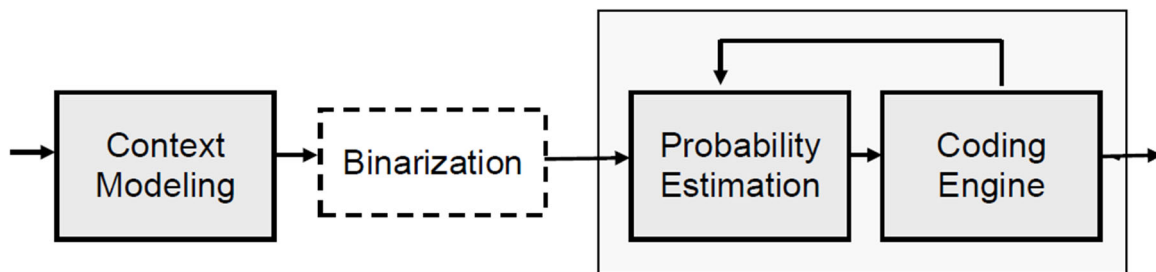
² The term “symbol” is commonly used to mean a unit of data (such as a character, sound sample, or pixel value) to keep the coding approach general so that it is applicable to various applications, such as text coding, audio coding, and/or image/video coding. I adopt that convention here.

length coding scheme. Tsai, 2:1-10.

54. Arithmetic coding operates by encoding symbols as intervals of real numbers between 0 and 1. Witten, 521. As more symbols are added to the message, the interval shrinks, and the number of bits required to specify the interval increases. Witten, 521. The more probable symbols reduce the interval less than less probable ones, thus requiring fewer bits to encode. Witten, 521.

55. Unlike Huffman coding, which requires each symbol to be encoded into an integer number of bits, arithmetic coding allows fractional bit usage. Witten, 521.

56. Context-Adaptive Binary Arithmetic Coding (“CABAC”) is a type of arithmetic coding introduced before the ’701 patent’s earliest priority date. *See* VCEG-L13; EX1001, 11:63-15:61. CABAC involves three main steps: context modeling, binarization, and adaptive arithmetic coding, as illustrated in Figure 1 of VCEG-L13 (below).



VCEG-L13, Figure 1.

57. Context modeling creates a statistical model based on previously encoded symbols, allowing the system to predict the probability of future symbols.

VCEG-L13, 1-2, 5. This model is chosen based on the context of the symbol being encoded and evolves as more data is processed. VCEG-L13, 5.

58. Binarization converts non-binary values (e.g., decimal numbers, such as integers) into sequences of binary decisions, known as “bins.” VCEG-L13, 1-2, 5. This process may follow a structured mapping using binary trees. VCEG-L13, 2, 5.

59. Arithmetic coding is then used to encode each bin, with the probability model derived from context modeling guiding the selection of the most efficient bit representation. VCEG-L13, 5.

60. Throughout the encoding process, the context model is continuously updated, allowing for rapid adaptation to the changing statistics of the video content. VCEG-L13, 2 (“[D]uring encoding the current context model will be updated with the previously encoded symbol. Hence, the model keeps track of the actual statistics and provides a fast adaptation.”) CABAC improves compression performance by adapting the binary arithmetic coding process based on the context of the data being encoded, reducing redundancies more effectively than prior art solutions, particularly at low and high bit rates. VCEG-L13, 6-7.

61. Before the ’701 patent’s earliest priority date, the Video Coding Experts Group (“VCEG”) under the International Telecommunication Union-Telecommunication Sector (“ITU-T”) had made significant progress in developing

CABAC as part of H.26L, which eventually became the H.264/Advanced Video Coding (“AVC”) compression standard.

VI. The ’701 Patent

62. The ’701 patent is titled “Method and System for Context-Based Adaptive Binary Arithmetic Coding” and issued on February 15, 2005, from Application 09/995,240 (“’240 application”), filed on November 27, 2001. The ’701 patent claims priority to Provisional Application 60/332,112, filed September 14, 2001. The ’701 patent relates to image and video compression using context-based arithmetic coding, and particularly, the coding of run and level values for the block of quantized DCT transform coefficients. EX1001, 1:10-13, 15:47-61.

A. Applicant-Admitted Prior Art (AAPA)

63. The ’701 patent includes a lengthy Background of the Invention section describing the prior art. EX1001, 1:17-15:61. As I discuss in further detail below, I agree that the technology described in the ’701 patent’s Background of the Invention section (referred to herein as “AAPA”) was known to a POSITA before the ’701 patent’s earliest priority date of September 14, 2001.

64. For example, the ’701 patent’s Background of the Invention explains that entropy coding was known in the art. EX1001, 2:59-3:18 (describing use of “entropy coding to further reduce the amount of data (i.e. number of bits) required”

in the “**widely-used JPEG standard**”),³ 6:32-52 (describing “[a] number of different methods” for “implement[ing] entropy coding”). I agree that entropy coding was known in the art before the ’701 patent’s earliest priority date.

65. The ’701 patent’s Background of the Invention explains that “[CABAC]...is a form of binary arithmetic coding which continually adapts to the statistical properties of the information being coded and **is known in the art** to be one of the most efficient forms of entropy coding[.]” EX1001, 11:63-12:6 (citing 1987 paper). The ’701 patent acknowledges that CABAC was introduced into “ITU-T recommendation H.26L.” EX1001, 11:37-39, 12:15-18. I agree that CABAC was known in the art before the ’701 patent’s earliest priority date.

66. The ’701 patent’s Background of the Invention describes a prior art generic hybrid video encoder and decoder with reference to FIGS. 1 and 2, respectively, that operate according to the H.26L standard. EX1001, 3:45-11:35, 11:43-47 (“The video encoding system defined by ITU-T H.26L...operates according to the general principles described...in connection with the generic video encoder 300 and decoder 400 illustrated in FIGS. 1 and 2.”). The ’701 patent describes “ITU-T recommendation H.26L” as “the latest in a family of video coding standards developed by the [ITU]...for video coding at very low bit rates[.]”

³ All emphases added unless otherwise stated.

EX1001, 11:37-40. As I discuss in § V above, the H.26L video coding standard was in development and known to a POSITA before the '701 patent's earliest priority date.

67. The '701 patent's Background of the Invention uses the generic hybrid video encoder and decoder of FIGS. 1 and 2 to "illustrate principles of block-based transform coding and motion-compensated prediction." EX1001, 4:25-31, 2:39-41 ("The basic approach to image compression used in almost all still image and video encoders existing today involves block-based transform coding."), 3:46-48 ("State of the art video coding systems make use of a technique known as 'motion-compensated prediction', to reduce the temporal redundancy in video sequences."). I agree that block-based transform coding and motion-compensated prediction were known in the art before the '701 patent's earliest priority date.

68. The '701 patent's "Background of the Invention" section describes run/level coding as known in the art. EX1001, 6:6-17, 6:28-32 ("Since certain values of levels and runs are more likely than other values to occur, entropy coding techniques can be used effectively to reduce the number of bits required to represent the run and level values."), 7:50-58, 9:50-60, 10:33-42. For example, the '701 patent explains that ITU-T H.26L uses run/level coding. EX1001, 12:7-26, 14:2-16:24, 16:62-64. I agree that run/level coding was known in the art before the '701 patent's earliest priority date.

69. The '701 patent's Background of the Invention indicates that during the compression process, the run and level values representing the transform coefficients are compressed using entropy coding, which uses statistics to reduce the amount of data used to store or transmit a signal. EX1001, 6:19-32. "[E]ntropy coding is often referred to as Variable Length Coding (VLC)" but different methods, such as look-up tables or arithmetic coding, can also be used to implement entropy coding. EX1001, 6:32-45. The '701 patent's Background of the Invention confirms that entropy coding was known in the art:

In still image compression, such as that performed according to the baseline mode of the widely-used JPEG standard, an image to be coded is first divided into an array of non-overlapping square blocks, each block comprising, for example, an 8×8 array of image pixels. In the case of the JPEG baseline, a two-dimensional Discrete Cosine Transform (DCT) is then applied independently to each of the image blocks. This has the effect of converting the image data from the pixel value domain to the spatial frequency domain and to produce a corresponding set of coefficient values, each of which is a weighting factor for a basis function of the two-dimensional DCT. The coefficient values thus produced are quantized and then coded in a lossless manner using entropy coding to further reduce the amount of data (i.e. number of bits) required for their representation. According to the JPEG baseline, the entropy coder employs only Huffman coding to produce a compressed bit-stream, although in other modes arithmetic coding may alternatively be used. Finally, data describing image and coding parameters (e.g. type of compression, quantization and coding tables, image size, etc.) is embedded in the bit-stream produced by the entropy encoder. As the JPEG standard comprises four

alternative coding modes and places few constraints on the quantization and coding tables that can be used, this is necessary in order to enable JPEG compressed bit-streams to be interchanged among different platforms and for images to be reconstructed without any ambiguity.

EX1001, 2:59-3:18; *see also*:

Such look-up tables are often defined by statistical analysis of training material comprising symbols identical to those to be coded and having similar statistical properties. An alternative technique, known as arithmetic coding, can also be used to convert the run and level values into variable length code-words. In arithmetic coding a group of symbols, for example the run and level values for a block of quantized transform coefficients, are coded as a floating point decimal number.

Once the run and level values have been entropy coded using an appropriate method, the video multiplex coder further combines them with control information, also entropy coded using a variable length coding method appropriate for the kind of information in question, to form a single compressed bit-stream of coded image information 335.

EX1001, 6:32-52.

70. AAPA notes that ITU-T recommendation H.26L is a family of video coding standards developed by the International Telecommunications Union intended for video coding at very low bit rates, typically below 64 kbits/s, suitable for the coding of digital video for transmission via radio communication networks or any fixed line communication network. EX1001, 11:36-44. In Test Model 8 (TML8) of the H.26L recommendation, the default mode for entropy coding is

Universal Variable Length Coding (UVLC), a look-up table method used to encode all syntax elements:

The latest version of the H.26L recommendation, known as Test Model 8 (TML8) and described in “H.26L Test Model Long Term Number 8 (TML-8) draft0” (ITU-T Telecommunications Standardization Section, Study Group 16, Video Coding Experts Group), specifies two alternative entropy coding modes. In the first (default) mode a so-called Universal Variable Length Coding (UVLC) method is used to encode all syntax elements. The UVLC coding mode is a look-up table method in which the same set of variable length code-words is used to represent all the different kinds of information produced by the video encoder, regardless of the type of information in question.

EX1001, 11:52-63.

71. AAPA notes that the alternative entropy coding method is CABAC:

The alternative entropy coding method, specified for use in the so-called high complexity profile of H.26L, is a technique known as Context-based Adaptive Binary Arithmetic Coding (CABAC).

EX1001, 11:63-67.

72. AAPA points out that CABAC entropy coding was introduced to overcome the inherently sub-optimal nature of the UVLC entropy coding method, i.e., the statistical properties of the code-words do not match optimally with the characteristics of the information to be encoded:

Because UVLC entropy coding uses the same set of variable length code-words to represent all types of information produced by the video encoder, in general the

statistical properties of the code-words do not match optimally with the characteristics of the information to be encoded. For example, the frequency of occurrence of particular run and level values used to represent the quantized DCT coefficients for an INTRA-coded image block is likely to be different from the occurrence of values in control information relating to quantization parameter values. The CABAC entropy coding method was introduced into the H.26L recommendation in order to overcome the inherently sub-optimal nature of the UVLC entropy coding method.

EX1001, 12:7-18.

73. CABAC is “a form of binary arithmetic coding which continually adapts to the statistical properties of the information” to improve coding efficiency.

EX1001, 11:66-12:2.

Specifically, entropy coding methods which encode each symbol independently require at least one bit to represent each symbol. Because arithmetic coding represents groups of symbols with a single code-word, it is possible to achieve data compression rates of less than one bit per symbol. Thus the CABAC method provided in H.26L also provides the possibility of improved data compression. Furthermore, because it is an adaptive method, it is also able to take into account changes in the statistical characteristics of the information being coded, ensuring that data compression performance is maintained even if the nature of the data being encoded changes to some extent.

EX1001, 12:23-34.

74. AAPA explains how the CABAC method works:

CABAC is a binary arithmetic coding method and thus data symbols to be coded which have non-binary values

are first converted to binary values ('binarized') in binary mapping block 710. The binarization process involves mapping a symbol to be coded to a sequence of bins, each of which has a corresponding bin number and can be assigned a value of either 0 or 1... each of the bins is assigned to a so-called 'context' (hence the name context-based arithmetic coding). A context can be thought of as grouping together bins, which have similar statistical characteristics. In other words, each bin assigned to a particular context is assumed to have a similar probability of containing the value 1 or 0 as the other bins belonging to that context.

EX1001, 12:42-13:4.

B. Alleged Problem in the Prior Art and Purported Solution

75. The '701 patent alleges that the CABAC arithmetic encoding method in H.26L can be improved through the use of improved context models that account for specific relationships between run and level values associated with DCT transform coefficients. Those relationships are: "the level of a coefficient to be encoded has, in general, a magnitude substantially similar to the level of the previously encoded coefficient"; "larger level values are more likely to be preceded by smaller run values"; and "smaller level values are likely to be preceded by larger runs":

With specific reference to the CABAC method used in the high complexity profile of H.26L TML8, **the inventors of the present invention have determined that certain relationships exist between the run and level values associated with DCT transform coefficients.** They have further determined that these relationships can be used to construct improved context models which enable the

CABAC method to operate with improved coding efficiency when applied to the run and level values. In particular, the inventors have determined that **consecutive level values exhibit a significant similarity**. More specifically, within a given block of transform coefficients, **the level of a coefficient to be encoded has, in general, a magnitude substantially similar to the level of the previously encoded coefficient**. The inventors have also determined an inverse relationship between the level and run values. In particular, **larger level values are more likely to be preceded by smaller run values**. The converse is also true, namely **smaller level values are likely to be preceded by larger runs**. Consequently, the present invention proposes the creation of new context models for the coding of DCT transform coefficients, which take into account these relationships between level and run values.

EX1001, 16:2-24 (emphases added).

76. Based on such relationships, the '701 patent claims simply recite determining a probability characteristic (context) of a current level value based on a level value in a previous pair, and determining a context for a current run value based on the current level value. At most, the claims took pieces of what was already well known and combined them in obvious ways.

C. The Challenged Claims

77. A labelled list of the Challenged Claims is attached as **Appendix B** to this Declaration.

VII. CLAIM CONSTRUCTION

78. I understand from counsel that during an IPR proceeding, claims are to

be construed in light of the specification as would be read by a POSITA at the time the application was filed. I understand that claim terms are given their ordinary and customary meaning as would be understood by a POSITA in the context of the entire disclosure. A claim term, however, will not receive its ordinary meaning if the patentee acted as his or her own lexicographer and clearly set forth a definition of the claim term in the specification. In that case, the claim term will receive the definition set forth in the patent.

79. I have not been asked to construe any terms. Accordingly, I have given the '701 patent's claim terms their ordinary and customary meaning as would be understood by a POSITA in the context of the '701 patent's disclosure at the time the application for the '701 application was filed.

VIII. DETAILED EXPLANATION OF GROUNDS

Ground		Claims	Prior Art
1A	§103	1-14, 17-31, 34-40	Tsai, VCEG-L28
1B		15-16, 32-33	Tsai, VCEG-L28, VCEG-L13
2	§103	1-9, 18-26, 35, 37, 39	AAPA, Savatier
3	§103	10-17, 27-34, 36, 38, 40	AAPA, VCEG-L13

A. Ground 1A: Obviousness over Tsai and VCEG-L28 for Claims 1-14, 17-31, and 34-40.

1. Tsai

80. Tsai relates to “compression of run/level coded data.” Tsai, 1:12-15.

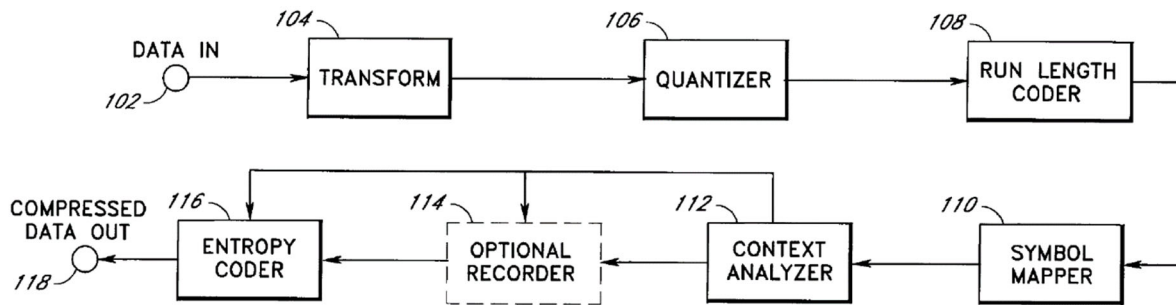
Run/level coded data is “a list of integers representing quantized data” that “can be described in terms of the size of runs of consecutive zeros, and the value or level of the nonzero coefficient that terminates each run of zeros,” represented as (run, level) pairs. Tsai, 1:50-52, 2:30-45. Tsai discloses a data compression system for input data such as image, video, or sound. Tsai, 6:60-64.

The digital data at input terminal 102 is input to a transform processor 104. The transform processor 104 generates transformed data which provide an alternative representation of the input data. The transform processor 104 can use one of a number of known transforms including, for example, Fourier, discrete cosine, or wavelet.

The transformed data is then input to a quantizer 106 coupled to the transform processor 104. The quantizer 106 maps the transformed data into a smaller set of output values. Quantization typically involves identifying a number of value ranges such that all data whose values lie within any particular range are replaced with a uniform value. Thus, for example, all values of transformed data between 0.5 and 1.5 might be replaced by 1.

Tsai, 7:1-18.

81. FIG. 1 (below) shows a data compression system in an embodiment. Tsai, 5:33-34, 6:60-61. Digital data is input to transform processor 104 to generate transformed data. Tsai, 7:1-4. The transformed data is subsequently passed to quantizer 106 and then to run-length coder 108. Tsai, 7:10-44.



Tsai, FIG. 1.

82. Tsai discloses run-length coder 108 in the data compression system that “orders the quantized integer data into a one-dimensional list of integers” and “converts this list of integers into a sequence of (run, level) pairs, where ‘run’ is a number representing a count of zeros occurring in consecutive order in the list of integers, and ‘level’ is a number representing the value of a nonzero integer occurring immediately after a series of consecutive zeros.” Tsai, 7:43-53. Tsai thus discloses techniques to compress the (run, level) pairs. I note that Tsai’s description of run-length (also called run/level) coding, and of “run” and “level” specifically, is consistent with the definitions of “level” and “run” known to a POSITA as of the ’701 patent’s earliest priority date. *See supra* § V.C.

83. Tsai explains that the manner in which the (run, level) pairs is mapped “plays a critical role in the performance of the compression method”:

The numbers in this list are defined as (run, level) pairs. In particular, the way in which this information is mapped for subsequent entropy coding plays a critical role in the

performance of the compression method. Previous methods have typically used a separate symbol to represent each (run, level) pair. The frequency of occurrence of all the symbols is tabulated, and is used in an entropy coding method in which frequently occurring symbols are represented using fewer bits.

What is needed to improve the art is a method whereby information represented as a list of integers, such as (run, level) pairs, can be substantially compressed.

Tsai, 2:44-55.

84. To improve the performance of entropy coding, Tsai discloses a specific mapping of (run, level) pairs to symbols and the use of two contexts: one for run, and one for level. Tsai, 4:6-5:9.

85. Tsai's mapping "eliminates explicit representation of the MSB for both run values and level values:"

The symbols that result from this mapping give increased coding efficiency for several reasons. First, the mapping process itself eliminates explicit representation of the MSB for both run values and level values. This reduces the number of output symbols, and contributes directly to greater efficiency.

Tsai, 4:39-43.

86. In addition, "the presence of two contexts ... increase[s] the performance of the entropy coding step":

Furthermore, the presence of two contexts is used to increase the performance of the entropy coding step that follows the mapping to symbols. Separate probability tables are developed for each context, and these

probability tables are used to design optimized entropy codes. If the statistics of the symbol occurrences in the two contexts changes, the entropy codes are adapted.

Tsai, 4:44-50.

87. Tsai explains: “It is ... desirable to perform the mapping such that the symbols comprising words appear with unequal probability. This will lead to greater efficiency in the entropy coding performed by the entropy coder 116.” Tsai, 10:42-45. For example, Tsai explains that “[t]o improve coding efficiency, it is desirable that shorter words be used to represent run values that have a higher probability of occurrence”: describes mapping the (run, level) pairs such that shorter words to represent run values that have a higher probability of occurrence, and to perform the mapping such that the symbols comprising words appear with unequal probability with examples illustrated in FIG. 5 of Tsai (reproduced below):

To improve coding efficiency, it is desirable that shorter words be used to represent run values that have a higher probability of occurrence. For typical image and sound data, the run-length coder 108 is more likely to produce runs that are small. Symbol Maps 1 and 2 (502, 504) in FIG. 5 takes advantage of this fact and associates small run values with short words.

It is also desirable to perform the mapping such that the symbols comprising words appear with unequal probability. This will lead to greater efficiency in the entropy coding performed by the entropy coder 116...

Tsai, 10:35-45.

502 SYMBOL MAP 1		504 SYMBOL MAP 2
RUN VALUE	WORD	WORD
1	+	+
2	-	-
3	++	++
4	--	+-
5	+-	--+
6	-+	---
7	+++	+++
8	---	++-
9	+-	+-+
10	-+-	-++
11	++-	+-
12	--+	-+-
13	+-+	---+
14	-++	---
15	++++	++++
16	----	++++-
.	.	.
.	.	.
.	.	.

Tsai, FIG. 5.

88. Similarly, Tsai discloses additional efficiency in entropy coding can be gained when input symbols appear with unequal probability, as shown in FIG. 7:

FIG. 7 illustrates two symbol maps which could be used to assign words to level values. Symbol Map 1 702 represents the symbol mapping strategy of FIG. 6 wherein words are assigned to level values in a binary ascending order. Symbol Map 2 704, however, represents an alternate map for associating words with level values. As explained above with respect to symbol mapping for run values, additional efficiency in entropy coding can be gained when input symbols appear with unequal probability. Within any group of level words having the same length, Symbol Map 2 arranges the words so that the

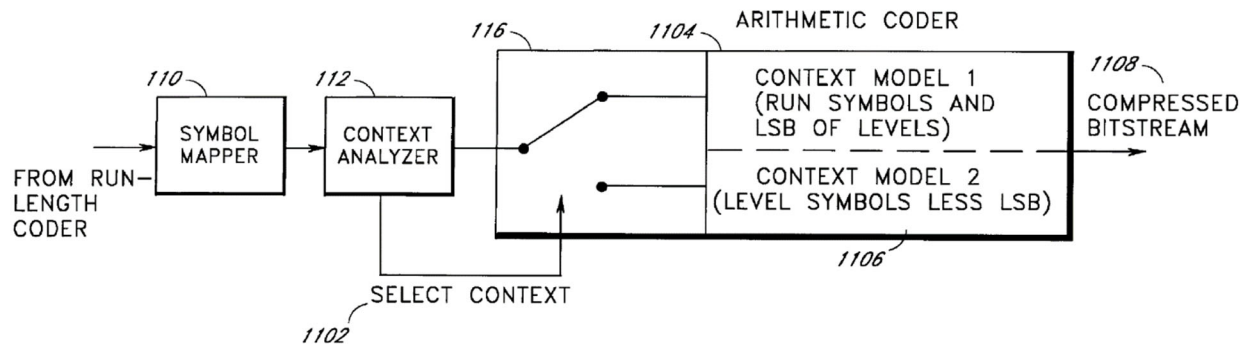
symbol “0” occurs more frequently in words associated with small level values. Thus, in the case where run/level coded data contain a large percentage of low level values, better compression ratios could be expected by using Symbol Map 2 704 instead of Symbol Map 1 702.

Tsai, 11: 46-61.

ABSOLUTE VALUE OF LEVEL	SYMBOL MAP 1		SYMBOL MAP 2	
	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL
1	0+	0-	0+	0-
2	1+	1-	1+	1-
3	00+	00-	00+	00-
4	10+	10-	10+	10-
5	01+	01-	01+	01-
6	11+	11-	11+	11-
7	000+	000-	000+	000-
8	100+	100-	100+	100-
9	010+	010-	010+	010-
10	110+	110-	001+	001-
11	001+	001-	110+	110-
12	101+	101-	101+	101-
13	011+	011-	011+	011-
14	111+	111-	111+	111-
15	0000+	0000-	0000+	0000-
⋮	⋮	⋮	⋮	⋮

Tsai, FIG. 7.

89. FIG. 11 of Tsai (reproduced below) shows the flow of data from the symbol mapper 110 into the context analyzer 112 and then into the arithmetic coder 116.

**Tsai, FIG. 11.**

90. Tsai further discloses in Figure 11 that after the symbol mapper 110 maps the (run, level) pairs into a symbol stream, the context analyzer 112 partitions the symbol stream into two different types of symbol groups by breaking up the symbol stream into groups of symbols each of which is slightly different than a run word or a level word:

The context analyzer 112 performs two functions. First, the context analyzer 112 partitions the symbol stream into two different types of symbol groups. The context analyzer 112 performs this partitioning by breaking up the symbol stream into groups of symbols each of which is slightly different than a run word or a level word. The second function performed by the context analyzer 112 controls the context of arithmetic coding. Both functions of the context analyzer are described more fully as follows.

Tsai, 15:7-15.

91. Tsai teaches that superior compression levels can be obtained “using an arithmetic coder 116 (FIG. 11) capable of maintaining two context models during coding—one context model 1104 for coding a first symbol stream and a second

context model 1106 for coding a second symbol stream, where the first and second symbol streams have different symbol frequencies.” Tsai, 15:60-66.

92. FIG. 12 of Tsai (reproduced below) shows an example partitioning of the input symbol stream into two types of symbol groups, where a location list represents “[t]he first type of symbol group [that] consists of a run word and the least significant bit (LSB) of the following level word,” and a level list represents “[t]he second type of symbol group ... [that] can be described simply as level words without a LSB.” Tsai, 15:18-20, 38-40.

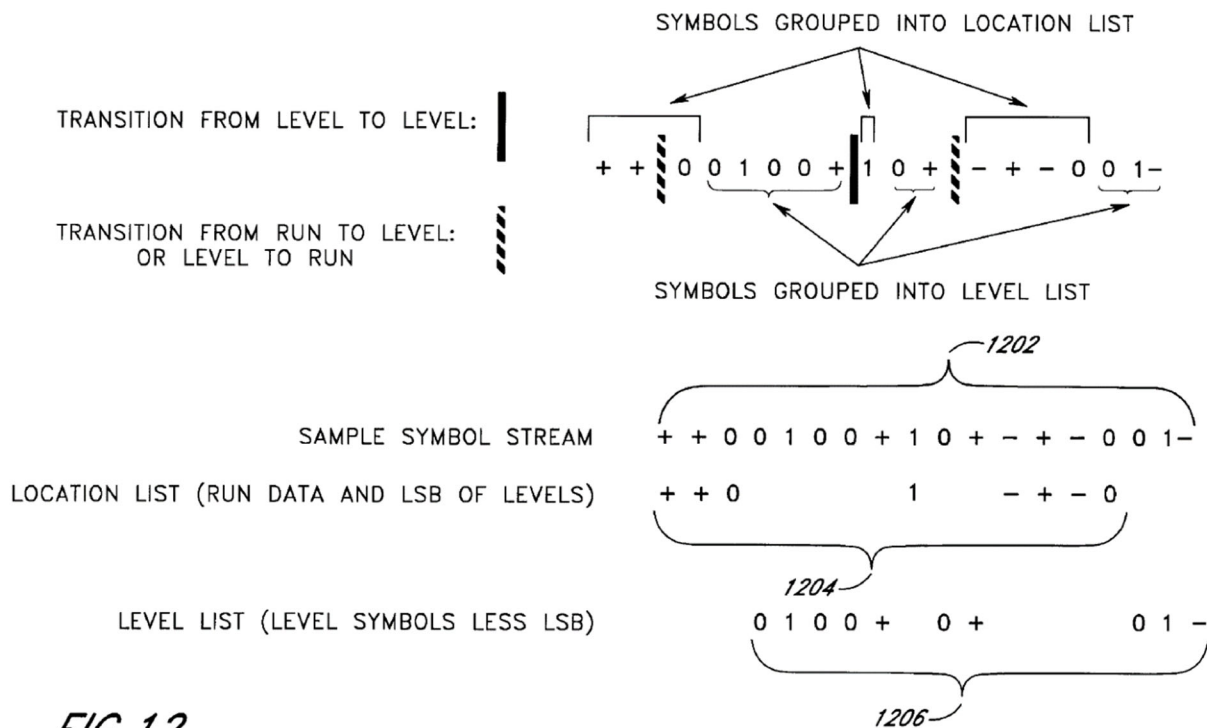


FIG. 12

Tsai, FIG. 12.

93. Tsai teaches that superior compression levels can be obtained “using an arithmetic coder 116 (FIG. 11) capable of maintaining two context models during

coding—one context model 1104 for coding a first symbol stream and a second context model 1106 for coding a second symbol stream, where the first and second symbol streams have different symbol frequencies.” Tsai, 15:60-66. “By partitioning the input symbol stream into two types of symbol groups, the invention takes advantage of the different symbol frequencies associated with each type of symbol group. The context-sensitive arithmetic coder 116 can thus achieve greater compression by coding symbols from each of the two groups using two different adaptive context models (1104, 1106).” Tsai, 16:19-26.

2. VCEG-L28

94. VCEG-L28 is a proposal from the Video Coding Experts Group (“VCEG”) under the International Telecommunication Union-Telecommunication Sector (“ITU-T”) for modifying the H.26L standard. VCEG-L28,1. On its face, VCEG-L28 indicates that it was published on January 3, 2001 for discussion during January 9-12, 2001. VCEG-L28, 1.

95. In the H.263 standard, the transform coefficients are coded using variable length coding (VLC). H.263, § 5.4.2, Table 13. H.26L, as an improvement of H.263, uses a universal VLC (UVLC) table to code all syntax elements. TML4, § 2.4.1. However, “for intra coefficients and specially for small QP values the UVLC is not optimal because the statistics of the events are not very suitable for the UVLC.” Q15-K-30, 1. A second VLC can be used to address such issues. Q15-K-

30, 1. Coding efficiency can be improved by automatically switching between the two VLCs that are designed to better fit the event statistics. Q15-K-30, 2.

96. VCEG-L28 also recognizes that coding improvements can be achieved by switching between two VLC tables and accordingly presents techniques for a new VLC approach consistent with H.26L. VCEG-L28, Background. VCEG-L28 presents a second UVLC2 table. The first nine codewords of UVLC and UVLC2 are shown in Table 1 (reproduced below). By using UVLC2 when coding a symbol, up to two bits can be saved to achieve better coding performance. VCEG-L28, §1. VCEG-L28 teaches that “coding improvements are possible by switching between two VLC tables” during the run/level coding process. VCEG-L28, §1. Each table corresponds to codewords associated with certain event statistics. VCEG-L28, §§1-2, 4. For example, Table 1 of VCEG-L28 (below) shows the first nine codewords of UVLC and UVLC2.

Table 1. The first nine codewords of UVLC and UVLC2

Index	UVLC	UVLC2
0	1	11
1	001	001
2	011	011
3	00001	1001
4	00011	1011
5	01001	00001
6	01011	00011
7	0000001	01001
8	0000011	01011

VCEG-L28, Table 1.

97. In VCEG-L28, switching between the two tables, UVLC and UVLC2, can occur based on the following rules:

4. Switching between UVLC and UVLC2

The switching applies to Intra luma coefficients in double scan with the following rules:

- First coefficient, first scan: use UVLC2 if the above or left block is unavailable or if $\text{abs}(\text{dc_above}) + \text{abs}(\text{dc_left}) > 2$, where $\text{dc}=1$ is used outside the present macroblock.
- First coefficient, second scan: use UVLC2 if $\text{abs}(\text{dc}) > 2$ or if first scan had >3 coefficients.
- Second coefficients: use UVLC2 if $\text{abs}(\text{previous_level}) > 2(1)$ for first (second) scan.
- Third through sixth coefficients: use UVLC2 if $\text{abs}(\text{previous_level})/(\text{previous_run}+2) > 0$.

VCEG-L28, §4.

98. VCEG-L28 shows that switching to UVLC2 when appropriate can bring coding improvements as compared to H.26L.

5. Coding improvements

The coding improvements are reported in the included Excel sheet, where quantizers 10, 13, 16, 19, 22 and 25 have been used. The bit reduction figures were computed for each rate-distortion point of the old TML curves by comparing with the linearly interpolated bit usage of the new curves at the same PSNR values.

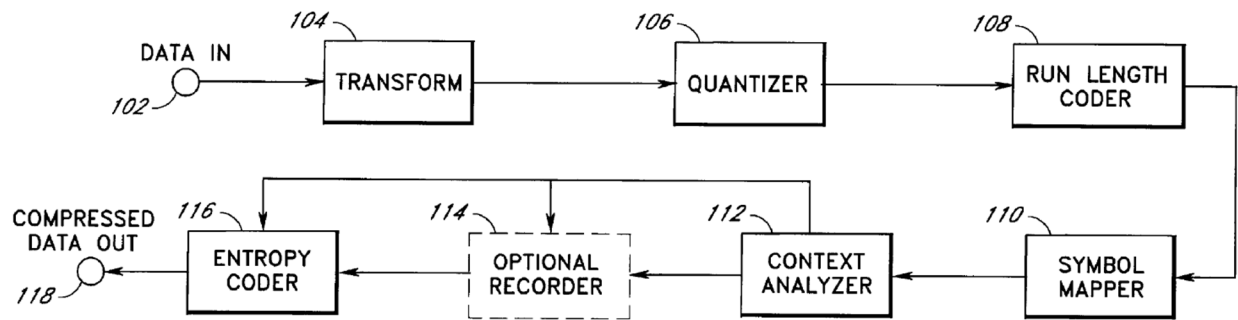
It is also possible to construct another set of codewords, UVLC3, consisting of original UVLC codewords with zero, one or two prefixes. By switching between UVLC, UVLC2 and UVLC3 even higher gains can be achieved. The corresponding figures are included in the chart as information.

VCEG-L28, §5.**3. Claim 1****a. 1[pre]**

“A method of image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values being scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, the method comprising the steps of:”

99. In my opinion, to the extent 1[pre] is limiting, Tsai discloses it.

100. Tsai relates to “data compression” in the context of the JPEG and MPEG compression standards, each of which a POSITA would have understood to be *a method of image coding*. Tsai, 1:13-15, 1:62-2:22. Tsai’s FIG. 1 (below) shows “a data compression system” that “can compress many types of data...for example, image, video, or sound.” Tsai, 6:60-67. A POSITA would have thus understood Tsai’s data compression system to be *an encoder* that employs *a method of image coding* because it encodes data such as image and video data.



Tsai, FIG. 1

101. Transform processor 104 of Tsai's data compression system performed the DCT operation on "8x8 pixel blocks comprising an image." Tsai, 7:1-9; *see id.*, 1:63-2:22, 4:6-23. Tsai thus discloses *an image is divided in an encoder into a plurality of blocks having a plurality of pixels.*

102. Tsai also recognizes that the pixels are represented using pixel values. Tsai, 2:34-37 ("Many lossless coding methods operating on digital data also involve the creation of lists of integers. These lossless methods work by using a predictor to estimate **the value of each pixel.**" (emphasis added)).

103. Tsai explains that DCT is "one of a number of known transforms" performed by "transform processor 104 [to] generate[] transformed data which provide an alternative representation of the input data." Tsai, 7:2-6. The '701 patent acknowledges that DCT is a transform coding operation. EX1001, 2:57-3:3.

104. The DCT operation produces "transformed data values...referred to as coefficients." Tsai, 2:3-4. As shown in FIG. 4A of Tsai (below), the transformed data values are represented as a block of coefficients. Tsai, 9:52-54. A POSITA

would have understood that the DCT operation transforms each 8x8 block of pixel values into a corresponding 8x8 block of DCT coefficients. Riley, 22 (“The DCT converts a block of pixels into a block of transform coefficients (of the same dimensions).”). Tsai thus teaches “*a transform coding operation [DCT] is performed on a block of pixels to produce a corresponding block of transform coefficient values.*”

FIG. 4A

402

0	0	0	35
4	0	0	0
0	0	0	0
0	0	0	-11

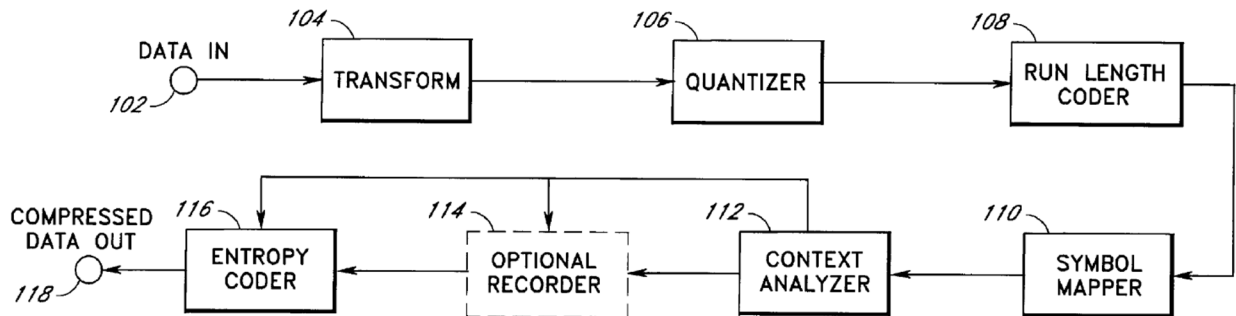
INPUT DATA ARRAY

Tsai, FIG. 4A

105. Tsai explains that transform processor 104 outputs transformed data corresponding to an 8x8 pixel block (*the block of transform coefficient values*) to quantizer 106, which “maps the transformed data into a smaller set of output values.” Tsai, 7:10-13. “[T]he quantized data consist of positive and negative integers, and

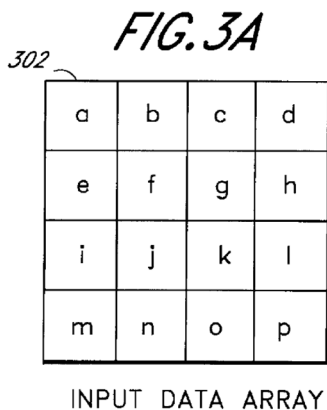
in some cases, a large percentage of these integers will be zero.” Tsai, 7:40-43.

106. As shown in Tsai’s FIG. 1 (below), “[t]he quantized data are input to...run-length coder 108.” Tsai, 7:43-44. Tsai explains “that transformation and quantization performed in respective sequence may yield data containing relationships in two or more dimensions.” Tsai, 9:25-29. As a result, “a first function of the run-length coder 108 is to map the data into a one-dimensional list.” Tsai, 9:29-31, 7:46-47.



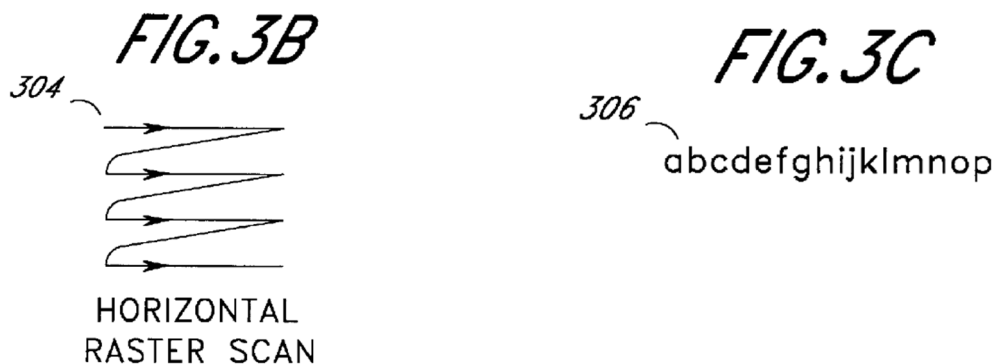
Tsai, FIG. 1.

107. “FIG. 3a [below] shows a two-dimensional data array 302 arranged in a 4×4 matrix.” Tsai, 9:34-35. Tsai explains that “two-dimensional data array 302...can be scanned in at least three different ways to obtain a one-dimensional data array.” Tsai, 9:36-38.



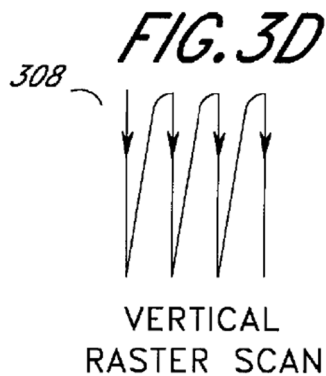
Tsai, FIG. 3A.

108. First, run-length coder 108 can use “horizontal scanning pattern 304 of FIG. 3b to obtain the data string 306 of FIG. 3c.” Tsai, 9:37-40.



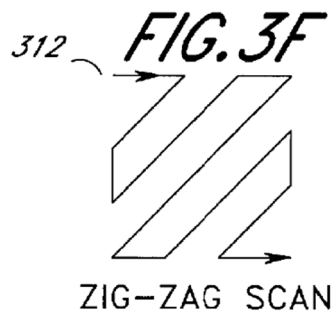
Tsai, FIGS. 3B-3C.

109. Second, run-length coder 108 can use “vertical scanning pattern 308 of FIG. 3d to obtain the data string 310 of FIG. 3e.” Tsai, 9:39-41.



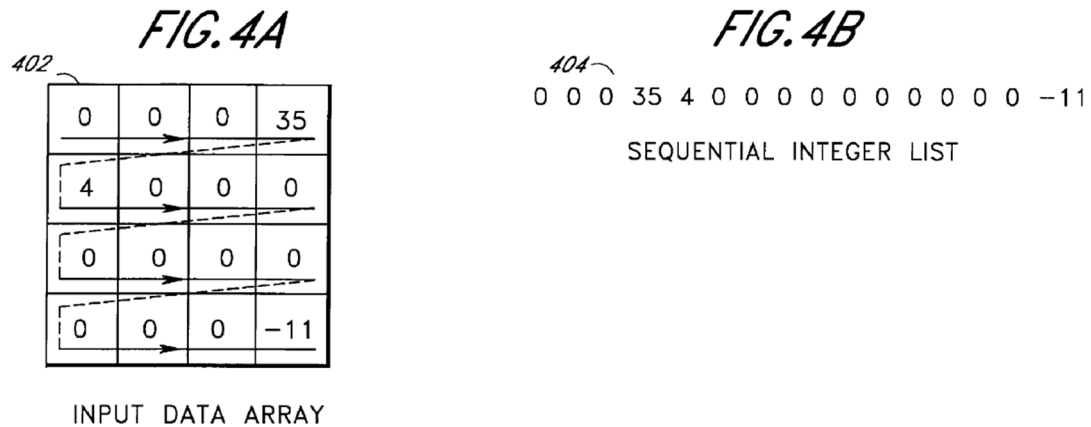
Tsai, FIGS. 3D-3E.

110. Third, run-length coder 108 can use “zig-zag scanning pattern 312 of FIG. 3f to obtain the data string 314 of FIG. 3g.” Tsai, 9:41-43.



Tsai, FIGS. 3F-3G.

111. Furthermore, FIGS. 4A-4B of Tsai demonstrate the generation of a scanned array of coefficients. FIG. 4A (below, left) shows “a two-dimensional array 402 of integers as might be output by a quantizer.” Tsai, 9:52-54. FIG. 4B (below, right) shows “the resulting sequential list of integers 404” after “[a]pplying the horizontal scan pattern 304 (FIG. 3b) to the two-dimensional array 402.” Tsai, 9:54-56. As shown, the resulting list is arranged according to the order of the horizontal scanning operation.



Tsai, FIGS. 4A-4B.

112. Tsai thus teaches scanning *the block of transform coefficient values* output by the *transform coding [DCT] operation ... in a given scanning order* (horizontal, vertical, zig-zag) *to produce a scanned array of coefficient values arranged according to the scanning order.*

b. 1[a]

“representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number; and”

113. In my opinion, Tsai discloses this limitation. After ordering the quantized integer data into a one-dimensional list of integers as discussed for 1[pre], Tsai’s “run-length coder 108 converts this list of integers into a sequence of (run, level) pairs.” Tsai, 7:48-49; *see id.*, 2:39-44.

114. FIG. 4D of Tsai (below) discloses the coefficient values arranged according to the scanned order are represented as run/level pairs, each having a level

value (*a first number*) and a run value (*a second number*). Tsai, 10:5-9.

(3,35) (0,4) (10,-11)
RUN/LEVEL PAIRS

Tsai, FIG. 4D.

115. I note that in H.263, the run values are positioned before the level values in combinations of (LAST, RUN, LEVEL), where LAST indicate whether there are more non-zero coefficients in the block. H.263, §5.4.2, Table 13. Like H.263, Tsai also positions run values before the level values, as shown in FIG. 4D of Tsai (above).

116. In H.26L, however, the run values can be positioned after the level values, and sometimes in the format of LEVEL, RUN, and End Of Block (EOB). TML4, Table 1, § 4.3.4.

117. The '701 patent uses both arrangements for level and run values. *See* EX1001, FIG. 6b. I note that Claims 9, 17, 26, and 34 of the '701 patent recite *said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value*. I thus interpret *a first number* as a level value and *a second number* as a run value.

c. 1[b]

“assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair.”

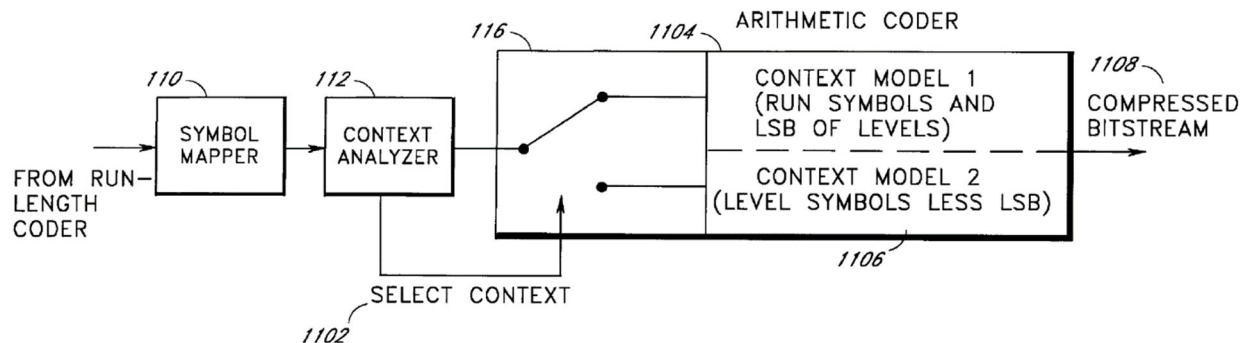
118. In my opinion, Tsai discloses this limitation. As discussed above for 1[a], in Tsai, DCT coefficient values are represented as run/level pairs, each having a level value (*a first number*) and a run value (*a second number*). Tsai discloses:

Furthermore, the presence of two contexts is used to increase the performance of the entropy coding step that follows the mapping to symbols. Separate probability tables are developed for each context, and these probability tables are used to design optimized entropy codes. If the statistics of the symbol occurrences in the two contexts changes, the entropy codes are adapted.

Tsai, 4:44-50. In other words, Tsai teaches that different statistical characteristics correspond to different probability tables used during the entropy coding step.

119. Tsai’s FIG. 11 (below) presents a block diagram that outlines how data flows from the symbol mapper to the context analyzer and then to the arithmetic coder. Tsai, 14:67-15:6. The process begins with the symbol stream generated by the symbol mapper 110. This stream consists of symbols representing either run values or level values. The context analyzer 112 takes in this symbol stream and examines the symbols to determine their context, specifically whether they represent “run” or “level” values. This distinction is crucial because run values and level values are encoded using different alphabetic symbols, and the context analyzer

identifies these differences to properly interpret the data.



Tsai, FIG. 11.

120. Once the context analyzer 112 has determined the type of symbols it is processing, it prepares the data for the arithmetic coder, which is responsible for further compressing the symbol stream by efficiently encoding the symbols based on their frequency of occurrence. By detecting the appropriate context, the context analyzer 112 ensures that the data is handled correctly during the compression process, which ultimately improves the overall efficiency of the encoding system. Tsai, 14:67-15:6.

121. Tsai further discloses (with reference to FIG. 11) that when the context analyzer 112 determines “a group of symbols comprising **a level word** with no LSB,” it “issues a second select context command 1102 to the arithmetic coder 116” to cause “the arithmetic coder 116 to use the context model 1106 it is adapting to code groups of symbols consisting of **a level word** having no LSB.” Tsai, 16:12-18 (emphases added).

122. Therefore, Tsai discloses assigning the level words in the level group (*the first numbers*) to the level context of the two contexts (*one of a plurality of contexts representative of the first numbers*).

123. Furthermore, during the symbol mapping stage, Tsai discloses that “[i]t is also desirable to perform the mapping such that the symbols comprising words appear with unequal probability” to “lead to greater efficiency in the entropy coding.” Tsai, 10:42-45. In particular, “additional efficiency in entropy coding can be gained when input symbols appear with unequal probability.” Tsai, 11:53-54.

124. As shown in FIG. 7 of Tsai (below), two symbol maps corresponding to different probability statistics can be used by the symbol mapper. Tsai explains: “Symbol Map 2 arranges the words so that the symbol ‘0’ occurs more frequently in words associated with small level values. Thus, in the case where run/level coded data contain a large percentage of low level values, better compression ratios could be expected by using Symbol Map 2 704 instead of Symbol Map 1 702.” Tsai, 11:54-61. For example, when the absolute value of the level is 10 or 11 (annotated in red), the symbol representations for the level value are different depending on which Symbol Map is used. Symbol Map 2 uses two ‘0’ bins to map to level value 10 while Symbol Map 1 uses one ‘0’ bin to map to the same value. When the run/level coded data includes more low-level values such as value 10, more ‘0’ bins are included in the output stream, which can be better compressed as compared to

‘1’ bins.

ABSOLUTE VALUE OF LEVEL	SYMBOL MAP 1		SYMBOL MAP 2	
	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL
1	0+	0−	0+	0−
2	1+	1−	1+	1−
3	00+	00−	00+	00−
4	10+	10−	10+	10−
5	01+	01−	01+	01−
6	11+	11−	11+	11−
7	000+	000−	000+	000−
8	100+	100−	100+	100−
9	010+	010−	010+	010−
10	110+	110−	001+	001−
11	001+	001−	110+	110−
12	101+	101−	101+	101−
13	011+	011−	011+	011−
14	111+	111−	111+	111−
15	0000+	0000−	0000+	0000−
⋮	⋮	⋮	⋮	⋮

FIG. 7

Tsai, FIG. 7 (annotated).

125. In the same field of coding of run/level pairs of transform coefficients, VCEG-L28 discloses that switching between two tables, UVLC and UVLC2, can lead to improvements of coding efficiency. VCEG-L28, §2. Specifically, VCEG-L28 discloses that the level value in the previous value pair, *previous_level*, can be used to determine when to switch between UVLC and UVLC2 for symbol mapping. VCEG-L28, §4. A POSITA would have understood that VCEG-L28’s use of *previous_level*—i.e., the absolute value of the coefficient from a prior block or

scan—as an indicator to switch between different tables would allow the coder to better adapt to the nature of the data and lead to better compression of the data.

4. Switching between UVLC and UVLC2

The switching applies to Intra luma coefficients in double scan with the following rules:

- First coefficient, first scan: use UVLC2 if the above or left block is unavailable or if $\text{abs}(\text{dc_above}) + \text{abs}(\text{dc_left}) > 2$, where $\text{dc}=1$ is used outside the present macroblock.
- First coefficient, second scan: use UVLC2 if $\text{abs}(\text{dc}) > 2$ or if first scan had >3 coefficients.
- Second coefficients: use UVLC2 if $\text{abs}(\text{previous_level}) > 2(1)$ for first (second) scan.
- Third through sixth coefficients: use UVLC2 if $\text{abs}(\text{previous_level})/(\text{previous_run}+2) > 0$.

VCEG-L28, §4 (annotated).

126. By switching to UVLC2 when necessary for coding a symbol, VCEG-L28 explains that one bit is lost for index 0, but up to two bits can be saved when coding other index values, thereby leading to better coding efficiency. VCEG-L28, §2, Table 1. For example, as shown in Table 1 of VCEG-L28, index 3 is represented by four bits ‘1001’ in UVLC2 as compared to ‘00001’ in UVLC, thereby saving one bit. As another example, index 7 is represented by ‘01001’ in UVLC2 as compared to ‘0000001’ in UVLC, thereby saving two bits.

Table 1. The first nine codewords of UVLC and UVLC2

Index	UVLC	UVLC2
0	1	11
1	001	001
2	011	011
3	00001	1001
4	00011	1011
5	01001	00001
6	01011	00011
7	0000001	01001
8	0000011	01011

VCEG-L28, Table 1.

127. A POSITA looking to implement Tsai's teachings of the symbol mapper would have been motivated to incorporate VCEG-L28's teaching of switching between the two symbol tables based on the previous level value. A POSITA would have recognized that Tsai and VCEG-L28 relate to video coding systems, specifically, the run/level coding process, and both aim to improve video coding efficiency. Tsai, Abstract, 1:28-33, 2:53-55; VCEG-L28, Title, §5. VCEG-L28 and Tsai focus on optimizing symbol-based encoding, but they do so with different (but compatible) approaches. VCEG-L28 describes a method to improve intra-frame coding by switching between two Variable Length Code (VLC) tables (UVLC and UVLC2) based on specific context-driven conditions. Tsai's system similarly discloses a method to efficiently reduce the number of symbols required to represent data using two symbol maps (Symbol Map 1 and Symbol Map 2) for positive and negative values. A POSITA would have recognized the benefit in

combining VCEG-L28's teaching into Tsai's system to further optimize symbol representation based on the specific statistical properties of the data. Thus, a POSITA would have found it obvious that the benefits of VCEG-L28's coding techniques, which have the same goal as Tsai of improving coding efficiency, would have further improved the coding efficiency of Tsai's system.

128. A POSITA would have known that the context assignment of context in Tsai using the teachings of VCEG-L28 is based on the output symbol stream from Tsai's symbol mapper 110. Correspondingly, selection of the symbol table affects the output symbol stream and thereby also affects the context assignment. For example, run/level pairs of (3, 11) and (0, 10) can be represented as symbol stream “++001+110+” if Symbol Map 1 is used to represent both pairs. The resulting symbol stream leads to a partition of a location list (run data and LST of levels) that includes “++0” and “11,” and a level list (level symbols less LSB) that includes “01+” and “0+.” On the other hand, based on the level of the first pair (e.g., 11), the second run/level pair (0, 10) can be represented using Symbol Map 2, resulting a symbol stream of “++001+001+,” leading to an alternative partition of a location list that includes “++0” and “00,” and a level list that includes “01+” and “1+.” The alternative resulting symbol stream can cause the context analyzer and the arithmetic coder to adapt to the another specific model that corresponds to the mapped symbols.

129. A POSITA would have been motivated to take advantage of VCEG-

L28's context-based decisions for symbol mapping, improving compression efficiency and reducing computational overhead in Tsai's system. VCEG-L28's method is driven by context-specific conditions for switching between VLC tables, primarily targeting cases where larger absolute values of coefficients appear, which is determined by prior symbols or scan coefficients. Tsai's system, which utilizes Symbol Map 1 and Symbol Map 2, provides a predefined mechanism for encoding positive and negative values but lacks the adaptive context-driven optimization that VCEG-L28 introduces.

130. A POSITA would have had a reasonable expectation of success in making the combination. VCEG-L28 highlights the need to maintain resynchronization properties in the code stream, meaning that switching between VLC tables should not introduce errors in decoding when there are bit errors. Tsai's system already operates with two separate symbol maps, which a POSITA would see as compatible with VCEG-L28's method of maintaining synchronization during adaptive switching. In fact, a POSITA would have recognized that UVLC and UVLC2 in VCEG-L28 closely correspond to Symbol Map 1 and Symbol Map 2 in Tsai such that the selection criteria employed in VCEG-L28 could easily be adopted for selection between Symbol Map 1 and Symbol Map 2 in Tsai's system to further improve Tsai's coding efficiency. A POSITA would have recognized that the combination uses a known technique (VCEG-L28's selection criteria) in a known

device ready for improvement (Tsai's encoder/decoder using Symbol Map 1 and Symbol Map 2 for context assignment), and the combination would yield predictable results (an encoder/decoder having enhanced overall system coding efficiency).

131. Therefore, the combination of Tsai and VCEG-L28 teaches that the assignment of context for the level value (*the first number of a first number pair*) is based on its mapped symbol stream, which is at least partly in dependence on the symbol table selection affected by the level value in a previous pair (*a first number of a second number pair*).

4. Claim 2

“A method according to claim 1, wherein the first number of the first number pair is assigned to a context at least partly in dependence on the first number of a preceding number pair in the scanning order.”

132. In my opinion, Tsai in view of VCEG-L28 discloses the additional limitations of this claim. As discussed in Ground 1A, 1[b], a POSITA would have been motivated to combine with Tsai VCEG-L28's teaching that, in the symbol mapping stage, the selection of the table can be based on *previous_level*, the level value of a preceding number pair. VCEG-L28, §4.

4. Switching between UVLC and UVLC2

The switching applies to Intra luma coefficients in double scan with the following rules:

- First coefficient, first scan: use UVLC2 if the above or left block is unavailable or if $\text{abs}(\text{dc_above}) + \text{abs}(\text{dc_left}) > 2$, where $\text{dc}=1$ is used outside the present macroblock.
- First coefficient, second scan: use UVLC2 if $\text{abs}(\text{dc}) > 2$ or if first scan had >3 coefficients.
- Second coefficients: use UVLC2 if $\text{abs}(\text{previous_level}) > 2(1)$ for first (second) scan.
- Third through sixth coefficients: use UVLC2 if $\text{abs}(\text{previous_level})/(\text{previous_run}+2) > 0$.

VCEG-L28, §4 (annotated).

133. Therefore, the combination discloses that the context of a level value is assigned (*the first number of the first number pair is assigned to a context*) at least partly in dependence on the level value in a previous pair in the given order (*the first number of a preceding number pair in the scanning order*).

5. Claim 3

“A method according to claim 1, wherein said plurality of contexts are contexts of a context-based arithmetic coder.

134. In my opinion, Tsai discloses the additional limitations of this claim. Tsai discloses a “context-sensitive arithmetic coder 116 [that] can [] achieve greater compression by coding symbols from each of the two groups using two different adaptive context models (1104, 1106).” Tsai, 16:23-26. FIG. 11 of Tsai (reproduced below) shows context models in a context-oriented arithmetic coder (*contexts of a context-based arithmetic coder*).

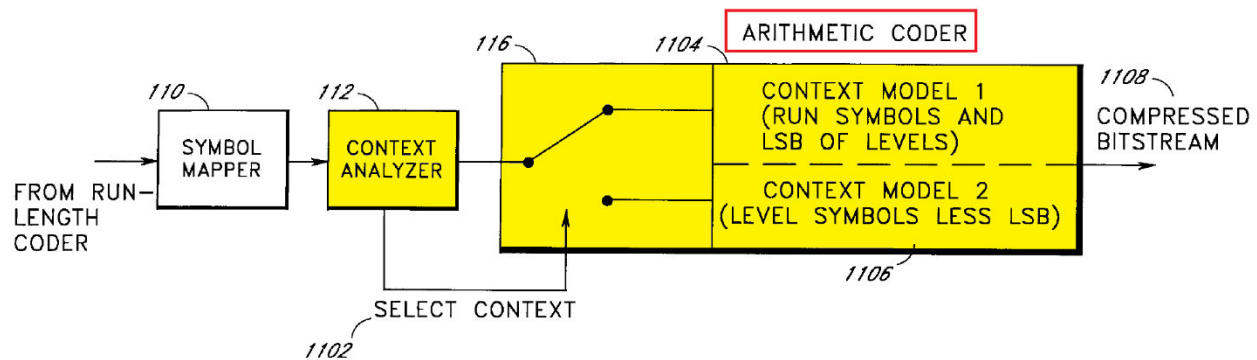


FIG. 11

Tsai, FIG. 11 (annotated).

6. Claim 4

“A method according to claim 1, wherein said plurality of contexts are contexts of a context-based binary arithmetic coder.”

135. In my opinion, Tsai discloses additional limitations of this claim. As discussed in Ground 1A, Claim 3, Tsai discloses a *context-sensitive arithmetic coder*. Tsai further takes advantage of “the unsigned binary representation of any number” and teaches that “the ‘0’ and ‘1’ symbols are assigned to words in a binary counting manner.” Tsai, 4:8-9, 10:66-67. A POSITA would have therefore understood Tsai’s arithmetic coder to be binary, and Tsai thus teaches *wherein said plurality of contexts are contexts of a context-based binary arithmetic coder*.

7. Claim 5

“A method according to claim 4, further comprising the step of: mapping the first numbers of said number pairs to a set of bins, each of which can be assigned one of either a first or a second

value.”

136. In my opinion, Tsai discloses the additional limitations of this claim. In mapping level values into symbols, Tsai teaches that “the ‘0’ and ‘1’ symbols are assigned to words in a binary counting manner.” Tsai, 10:66-11:2. The ’701 patent explains that “[t]he binarization process involves mapping a symbol to be coded to a sequence of bins, each of which has a corresponding bin number and can be assigned a value of either 0 or 1.” EX1001, 12:46-49. A POSITA would have understood that representing a value using the binary counting manner means that the value is represented as a set of binary values, or bins. Each binary value is 1 (*a first value*) or 0 (*a second value*).

8. Claim 6

“A method according to claim 5, wherein mapping of a number to a given one of said set of bins is indicated by assigning the bin to the first value.”

137. In my opinion, Tsai teaches the additional limitations of this claim. A POSITA would have understood that the binarization of a symbol is a process of mapping binary values, or bins, to a symbol. EX1001, 12:46-49. As shown in FIG. 7 of Tsai (below), the absolute values of levels are mapped in the binary format as a respective set of bins. For example, the value of 2 is mapped to a bin assigned to the value of 1 (*the first value*). Tsai therefore teaches mapping of 2 (*mapping of a number*) to a bin (*to a given one of said set of bins*) that is assigned the value of 1

(assigning the bin to the first value).

702		704		
ABSOLUTE VALUE OF LEVEL	SYMBOL MAP 1		SYMBOL MAP 2	
	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL
1	0+	0−	0+	0−
2	1+	1−	1+	1−
3	00+	00−	00+	00−
4	10+	10−	10+	10−
5	01+	01−	01+	01−
6	11+	11−	11+	11−
7	000+	000−	000+	000−
8	100+	100−	100+	100−
9	010+	010−	010+	010−
10	110+	110−	001+	001−
11	001+	001−	110+	110−
12	101+	101−	101+	101−
13	011+	011−	011+	011−
14	111+	111−	111+	111−
15	0000+	0000−	0000+	0000−
⋮	⋮	⋮	⋮	⋮

Tsai, FIG. 7 (annotated).

9. Claim 7

“A method according to claim 5, wherein the first value is 1 and the second value is 0.”

138. In my opinion, Tsai discloses the additional limitations of this claim. As discussed in Ground 1A, Claim 5, each binary value can be 1 (*the first value*) or 0 (*the second value*). For example, as shown in Tsai’s Table 7 (below), the level value 1 is assigned a bin value of “0” (yellow) and the level value 2 is assigned a bin value of “1” (green).

ABSOLUTE VALUE OF LEVEL	SYMBOL MAP 1		SYMBOL MAP 2	
	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL	WORD FOR POSITIVE LEVEL	WORD FOR NEGATIVE LEVEL
1	0+	0-	0+	0-
2	1+	1-	1+	1-
3	00+	00-	00+	00-
4	10+	10-	10+	10-
5	01+	01-	01+	01-
6	11+	11-	11+	11-
7	000+	000-	000+	000-
8	100+	100-	100+	100-
9	010+	010-	010+	010-
10	110+	110-	001+	001-
11	001+	001-	110+	110-
12	101+	101-	101+	101-
13	011+	011-	011+	011-
14	111+	111-	111+	111-
15	0000+	0000-	0000+	0000-
⋮	⋮	⋮	⋮	⋮

Tsai, FIG. 7 (annotated).

10. Claim 8

“A method according to claim 5, wherein the first number of a number pairs is assigned to a context at least partly in dependence on the bin to which the first number of said number pair is mapped.”

139. In my opinion, Tsai discloses the additional limitations of this claim.

As shown in FIGS. 10B-C of Tsai (below), the absolute value of a level (e.g., 35) is represented using multiple bins (e.g., ‘0’ ‘0’ ‘1’ ‘0’ ‘0’). Tsai, 14:39-43.

<i>FIG. 10B</i>		<i>FIG. 10C</i>	
RUN/LEVEL OUTPUT		SYMBOLS AFTER MAPPING (LSB-> MSB)	
1004	RUN	3	+ +
	LEVEL	35	0 0 1 0 0 +
	RUN	0	NOT CODED
	LEVEL	4	1 0 +
	RUN	10	- + -
	LEVEL	-11	0 0 1 -

Tsai, FIGS. 10B-10C (annotated).

140. The assignment of the context for a bin is at least partly based on the position of the bin in the multiple bins. Tsai, 15:16-57, Fig. 12 (reproduced below). For example, the first bin “0” (red) is grouped with the symbols “++” to be in the location list and be assigned the run context (blue). Tsai, 15:31-37. The second bin “0” (green) is grouped with the remaining bins to the level list and assigned the level context (orange). Tsai, 15:50-53.

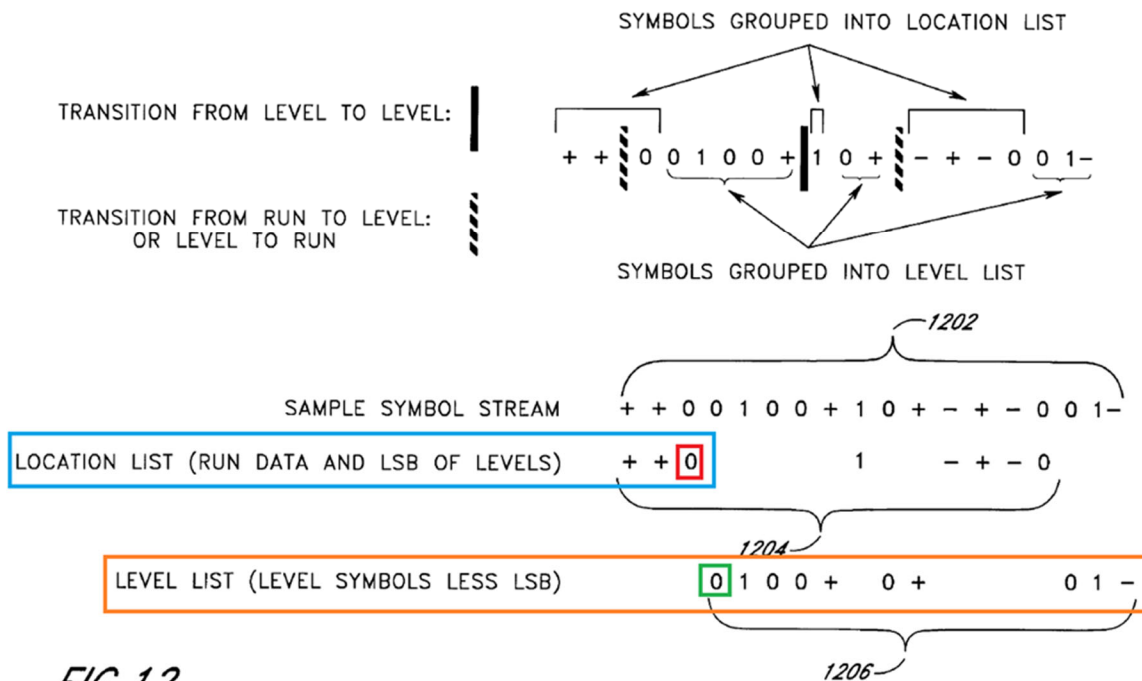


FIG. 12

Tsai, FIG. 12 (annotated).

141. Tsai therefore discloses that context assignment for a level (*the first number of a number pair[] is assigned to a context*) is based on the bin (e.g., the bin ‘0’) that the level is mapped to (*at least partly in dependence on the bin to which the first number of said number pair is mapped*).

11. Claim 9

“A method according to claim 1, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

142. In my opinion, Tsai discloses the additional limitations of this claim. Tsai discloses that the list with numbers defined as (run, level) pairs “can be described in terms of the size of runs of consecutive zeros, and the value or level of

the nonzero coefficient that terminates each run of zeros.” Tsai, 2:42-45.

143. As mentioned above, in H.263, the run values are positioned before the level values in combinations of (LAST, RUN, LEVEL), while in H.26L, the run values can be positioned after the level values, and sometimes in the format of LEVEL, RUN, and End Of Block (EOB). H.263, § 5.4.2, TML4, Table 1, § 4.3.4. I interpret *a first number* as a level value and *a second number* as a run value, which is consistent with the level value (*said first number*) indicating the value or level of the nonzero coefficient (*is indicative of a non-zero coefficient value*), and the run value (*said second number*) indicating the size of runs of consecutive zeros terminated by the non-zero level value (*is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value*).

12. Claim 10

a. 10[pre]

“A method of image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values is scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, the method comprising the steps of:”

144. To the extent this preamble is limiting, Tsai discloses this preamble for substantially the same reasons as discussed in Ground 1A, limitation 1[pre].

b. 10[a]

“representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number; and”

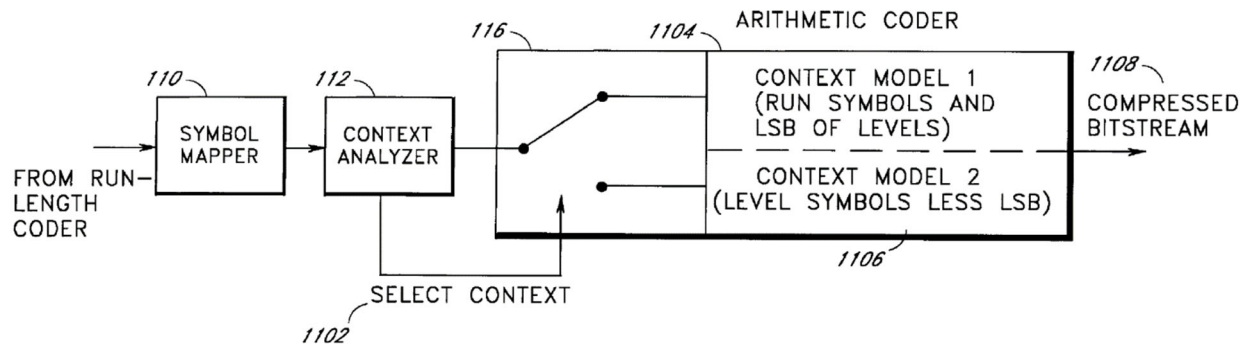
145. In my opinion, Tsai discloses 10[a] for substantially the same reasons as discussed in Ground 1A, 1[a].

c. 10[b]

“assigning the second numbers to one of a plurality of contexts representative of the second numbers such that the second number of a number pair is assigned to a context at least partly in dependence on the first number of the number pair.”

146. In my opinion, Tsai discloses this limitation. As discussed in Ground 1A, 1[b], Tsai teaches *a plurality of contexts* that includes one level context representative of the level words (*the first numbers*) and one run context representative of the run words (*the second numbers*).

147. FIG. 11 of Tsai (reproduced below) shows that, after partitioning the input symbol stream into two different types of symbol groups, the context analyzer “issues a select context command 1102 to the arithmetic coder 116” when it determines “a group of symbols comprising a run word and the LSB symbol of the following level word” to cause “the arithmetic coder 116 to use the context model 1104 which it is adapting to arithmetically code groups of symbols which comprise a run word and LSB symbol.” Tsai, 16:4-11. That is, the run words in the location/run group (*the second numbers*) are assigned to the run context.

**Tsai, FIG. 11.**

148. Tsai thus discloses assigning the run words in the run group (*assigning the second numbers*) to the run context of the two contexts (*to one of a plurality of contexts representative of the second numbers*).

149. FIG. 12 of Tsai (reproduced below) illustrates context analyzer 112 partitioning the input symbol stream 1202 into two different types of symbol groups, where the run values (*the second numbers*) are partitioned into the first type of symbol group that “consists of a run word and the least significant bit (LSB) of the following level word.” Tsai, 15:16-20.

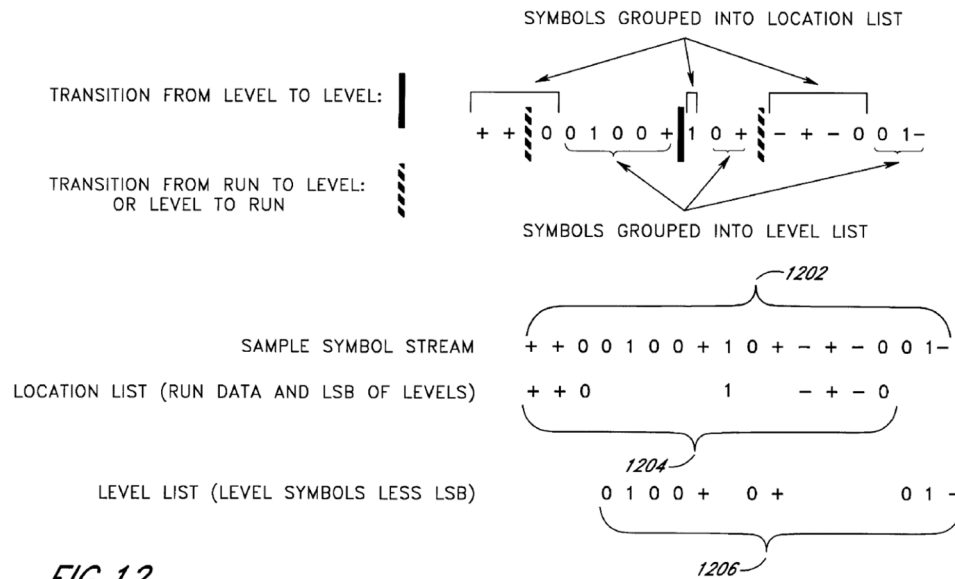


FIG. 12

Tsai, FIG. 12.

150. Tsai further discloses:

Instead of breaking the symbol stream 1202 at the end of the run word and passing the run word to the arithmetic coder 116, the context analyzer 112 adds the LSB of the following level word to the run word and passes this new group of symbols to the arithmetic coder 116. For example, when a run word of “++” and a level word of “00100+” occur in order (i.e., “++00100+”) in an input stream, as shown in FIG. 12, the context analyzer 112 will make a new location list symbol group by concatenating the run word “++” with the first (LSB) symbol of the following level word “0” to comprise the new symbol group “++0”.

The context analyzer 112 then “causes the arithmetic coder 116 to use the coding context model 1104 which it is adapting to arithmetically code groups of symbols which comprise a run word and LSB symbol [of the following level word].”

Tsai, 15:25-37, 16:6-11.

151. In other words, the assignment of the context to a run value is based on the partition that includes the LSB symbol of the level word.

152. Therefore, Tsai discloses that the run word (*the second number of a number pair*) is assigned to a context at least partly dependent on the LSB of the following level word in the pair (*the first number of the number pair*).

13. Claim 11

“A method according to claim 10, wherein said plurality of contexts are contexts of a context-based arithmetic coder.”

153. In my opinion, Tsai discloses Claim 11 for substantially the same reasons as discussed in Ground 1A, Claim 3.

14. Claim 12

“A method according to claim 10, wherein said plurality of contexts are contexts of a context-based binary arithmetic coder.”

154. In my opinion, Tsai discloses Claim 12 for substantially the same reasons as discussed in Ground 1A, Claim 4.

15. Claim 13

“A method according to claim 12, further comprising the step of: mapping the second numbers of said number pairs to a set of bins, each of which can be assigned one of either a first value or a second value.”

155. In my opinion, Tsai discloses additional limitations of this claim. Tsai notes a two-symbol alphabet that contains the symbols “+” and “-” for the run values. Tsai, 10:18-24. “Each run value then is associated with a symbol set ... [that

includes] a group of n symbols.” Tsai, 10:25-27. That is, each symbol can be considered as a binary value, or a bin, that can have either a value of “+” (*a first value*) or “-” (*a second value*).

16. Claim 14

“A method according to claim 13, wherein mapping of a number to a given one of said set of bins is indicated by assigning the bin to the first value.”

156. In my opinion, Tsai discloses the additional limitations of this claim. A POSITA would have understood that the binarization of a symbol is a process of mapping binary values, or bins, to a symbol. Tsai, 12:46-49. The binary values are not limited to ‘0’ and ‘1’ and can be other two-value sets. As shown in FIG. 5 of Tsai (below), a set of binary values – ‘+’ and ‘-’ – is used to represent run values. Run values are mapped to a respective set of bins, each having one of binary values of ‘+’ or ‘-.’ For example, the value of 1 is mapped to a bin assigned to the value of ‘+’ (*the first value*). As I explained for Ground 1A, Claim 5, Tsai’s mapping would have been understood by a POSITA to be an example of the common technique of binarization, wherein a symbol is mapped to binary values or bins. EX1001, 12:46-49.

502 SYMBOL MAP 1		504 SYMBOL MAP 2
RUN VALUE	WORD	WORD
1	+	+
2	-	-
3	++	++
4	--	+-
5	+-	-+
6	-+	--
7	+++	+++
8	---	++-
9	+--	+--
10	-+-	-++
11	++-	+-
12	--+	-+-
13	+-+	--+
14	-++	---
15	++++	++++
16	----	+++-
.	.	.
.	.	.
.	.	.

Tsai, FIG. 5.

157. Tsai thus discloses mapping the number 1 (*mapping of a number*) by assigning the binary value to the value of ‘+’ (*assigning the bin to the first value*).

17. Claim 17

“A method according to claim 10, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

158. In my opinion, Tsai discloses Claim 17 for substantially the same reasons as discussed in Ground 1A, Claim 9.

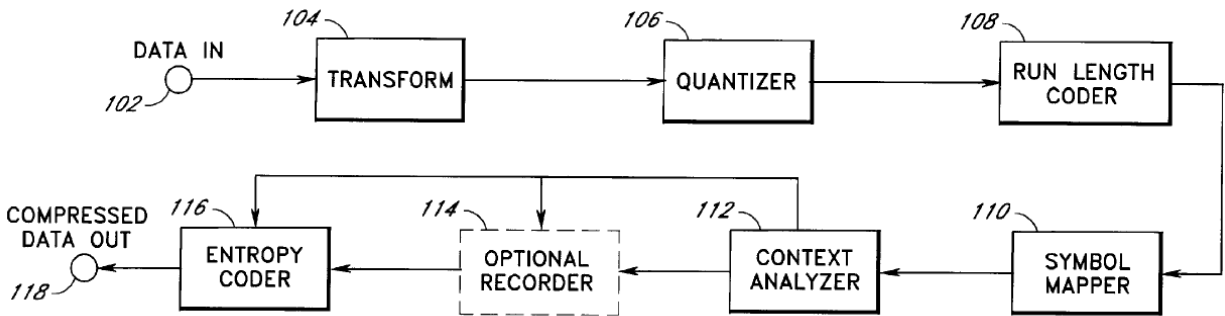
18. Claim 18

a. 18[pre]

“A system for image coding comprising:”

159. To the extent this preamble is limiting, Tsai discloses it. Tsai discloses coding methods in which a list of integers represents quantized data. Tsai, 2:39-40.

FIG. 1 of Tsai (reproduced below) illustrates a data compression system that can “compress many types of data such as, for example, image, video, or sound” and can be implemented to compress such lists substantially. Tsai, 6:61-64.



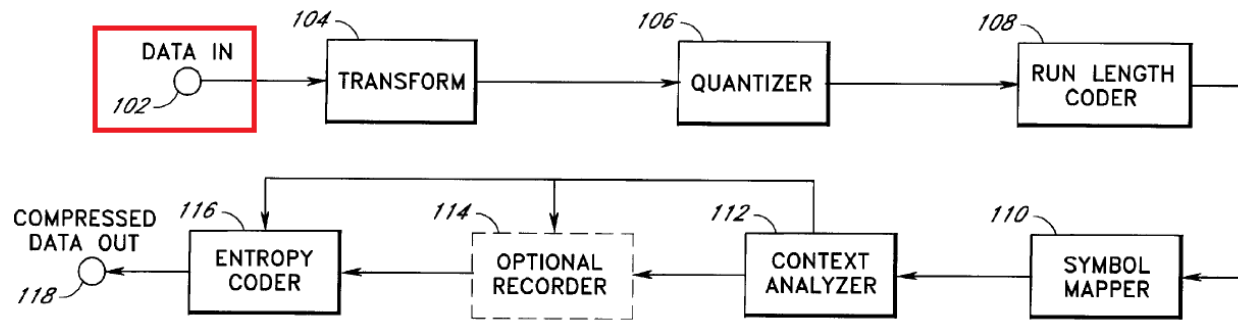
Tsai, FIG. 1.

160. Tsai thus discloses *a system for image coding*.

b. 18[a]

“an input for receiving an image as a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

161. In my opinion, Tsai discloses this limitation. Tsai discloses an input terminal 102 to receive data in a digital format of many types of data, such as an image. Tsai, 6:64-67. Tsai also teaches that an image can include image arrays, each having a block of pixels. Tsai, 13:46-47 (“an image array of size 8 pixels by 8 pixels”). A POSITA would have understood that each pixel has a pixel value. Tsai, 2:33-36.



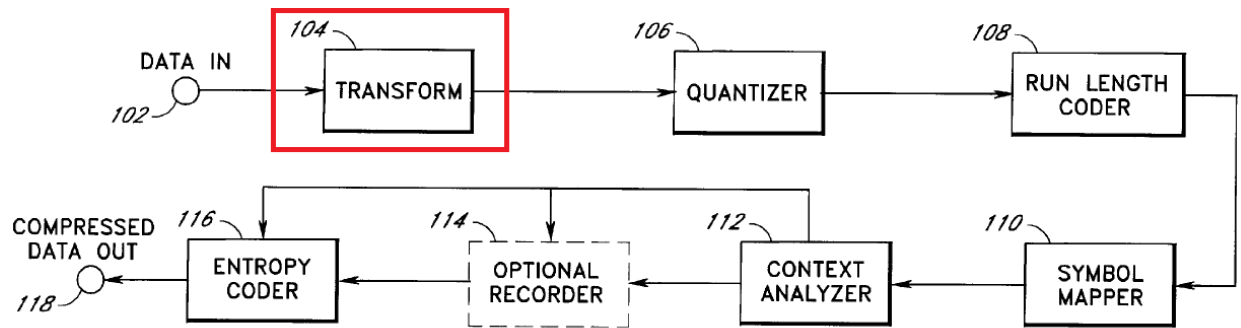
Tsai, FIG. 1 (annotated).

162. Tsai therefore teaches that the input terminal 102 (*an input*) receives an image having image arrays, each of size 8 pixels by 8 pixels (*receiving an image as a plurality of blocks having a plurality of pixels, each pixel having a pixel value*).

c. 18[b]

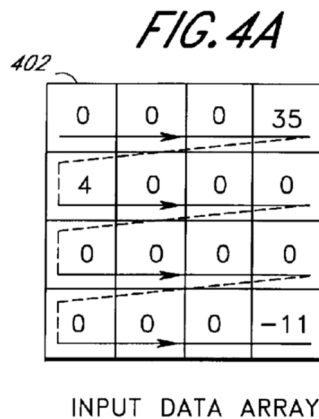
“a transform coder for performing a transform coding operation on a block of pixels to produce a corresponding block of transform coefficient values;”

163. In my opinion, Tsai discloses this limitation. Tsai discloses that “[t]he DCT, performed on 8x8 pixel blocks comprising an image, decorrelates the image data in that block. The transformed data values are referred to as coefficients.” Tsai, 2:1-4. Tsai’s system includes a transform processor 104 (*a transform coder*) that can use a number of known transforms including, for example, Fourier, discrete cosine, or wavelet, on 8x8 pixel blocks (*performing a transform coding operation on a block of pixels*) to produce transformed data values. Tsai, 7:4-6.



Tsai, FIG. 1 (annotated).

164. As shown in FIG. 4A of Tsai, the transformed data values are represented as a two-dimensional array of coefficients (*a corresponding block of transform coefficient values*). Tsai, 2:3-4.



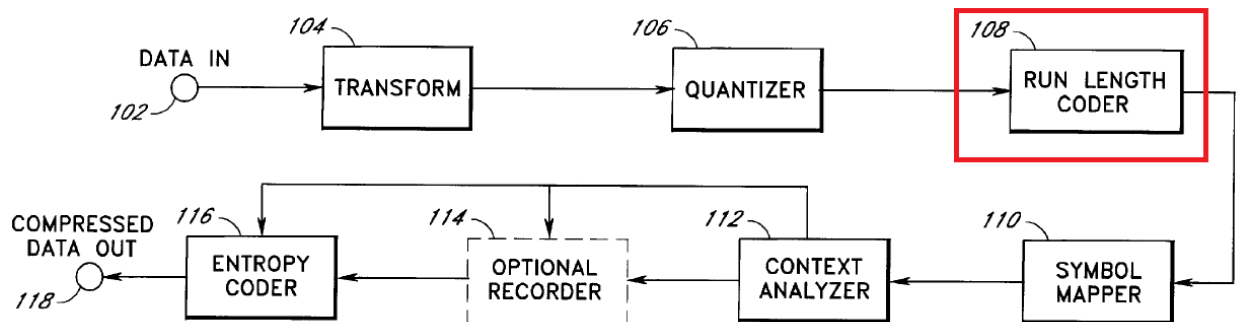
Tsai, FIG. 4A.

d. 18[c]

“a scanner for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values, arranged according to the scanning order;”

165. In my opinion, Tsai discloses this limitation. Tsai’s system includes a run-length coder 108 (*a scanner*) that maps the output from the transform processor

104 and the quantizer 106 into a one-dimensional list according to one of the many patterns in which two-dimensional data can be scanned (*scanning the block of transform coefficient values in a given scanning order*) to produce an array of coefficient values. Tsai, 9:30-34, Fig. 1 (below).



Tsai, FIG. 1(annotated).

166. As shown in FIGS. 4A-B (below), the array of coefficient values is arranged according to a given order (*a scanned array of coefficient values, arranged according to the scanning order*).

FIG. 4A

402

0	0	0	35
4	0	0	0
0	0	0	0
0	0	0	-11

INPUT DATA ARRAY

FIG. 4B

404

0 0 0 35 4 0 0 0 0 0 0 0 0 0 0 -11

SEQUENTIAL INTEGER LIST

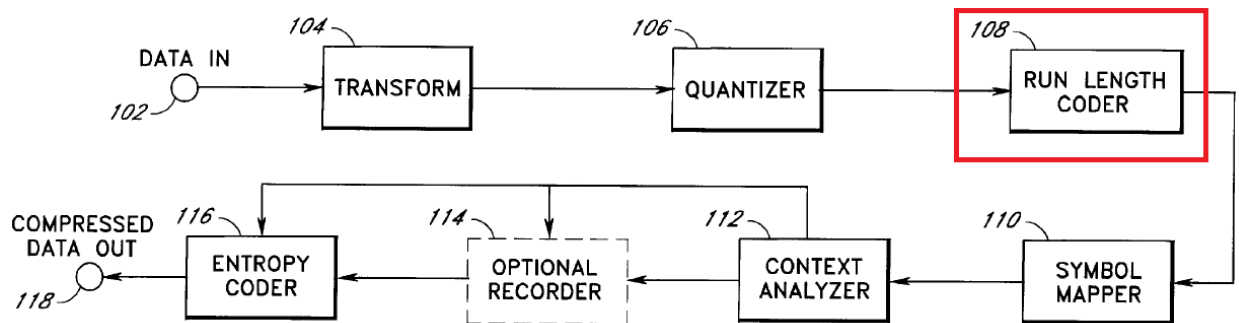
Tsai, FIGS. 4A-B.

e. 18[d]

“a run-level coder for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

167. In my opinion, Tsai discloses this limitation. In Tsai, the run-length coder 108 performs multiple functions, including “a first function...to map the data into a one-dimensional list” and a “second function...to convert a sequential list of integers into run/level output form.” Tsai, 9:29-31; 9:49-51. Tsai therefore combines the claimed *scanner* and *run-level coder* functionalities in a single component: run-length coder 108. It is worth noting that combining multiple functionalities in a single component was and is common practice. Tsai discloses

that run-length coder 108 (*a run-level coder*) converts a sequential list of integers into the run/level output form (*representing the coefficient values in the scanned array by a plurality of number pairs*). Tsai, 9:49-51. For example, the coefficient values can be represented as run/level pairs, e.g., (3, 35) (0, 4) (10, -11), each having a level value (*a first number*) and a run value (*a second number*). Tsai, 10:5-9.



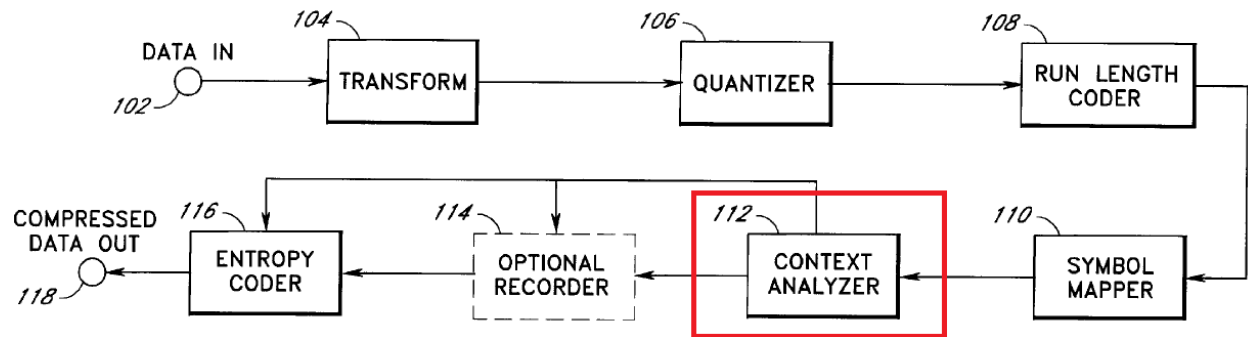
Tsai, FIG. 1 (annotated).

f. 18[e]

“a context-based coder for assigning the first numbers to one of a plurality of contexts representative of the first numbers and operative to assign the first number of a first number pair to a context at least partly in dependence on a first number of a second number pair.”

168. In my opinion, Tsai discloses this limitation. Tsai’s system includes a context analyzer 112 (*a context-based coder*) that issues a select context command 1102 for a level list to cause the arithmetic coder 116 to use the context model 1106 it is adapting (*assigning ... one of a plurality of contexts*) to code groups of symbols consisting of a level word having no LSB (*the first numbers*). Tsai, 16:12-18. Tsai discloses the remaining features of 18[e] for substantially the same reasons as

discussed in Ground 1A, 1[b].



Tsai, FIG. 1 (annotated).

19. Claim 19

“A system according to claim 18, wherein said context-based coder is a context-based arithmetic coder.”

169. In my opinion, Tsai discloses Claim 19 for substantially the same reasons as discussed in Ground 1A, Claim 3.

20. Claim 20

“A system according to claim 18, wherein said context-based coder is a context-based binary arithmetic coder.”

170. In my opinion, Tsai discloses Claim 20 for substantially the same reasons as discussed in Ground 1A, Claim 4.

21. Claim 21

“A system according to claim 20, wherein the context-based coder is operative to map the first numbers of said number pairs to a set of bins, each of which can be assigned to one of either a first or a second value.”

171. In my opinion, Tsai discloses Claim 21 for substantially the same reasons as discussed in Ground 1A, Claim 5.

22. Claim 22

“A system according to claim 21, wherein the context-based coder is operative to map a number to a given one of said set of bins by assigning the bin to the first value.”

172. In my opinion, Tsai discloses Claim 22 for substantially the same reasons as discussed in Ground 1A, Claim 6.

23. Claim 23

“A system according to claim 21, wherein the first value is 1 and the second value is 0.”

173. In my opinion, Tsai discloses Claim 23 for substantially the same reasons as discussed in Ground 1A, Claim 7.

24. Claim 24

“A system according to claim 21, wherein the context-based coder is operative to assign the first number of a number pair to a context at least partly in dependence on the bin to which the first number of said number pair is mapped.”

174. In my opinion, Tsai discloses Claim 24 for substantially the same reasons as discussed in Ground 1A, Claim 8.

25. Claim 25

“A system according to claim 18, wherein the context-based coder is operative to assign the first number of the first number pair to a context at least partly in dependence on the first number of a preceding number pair in the scanning order.”

175. In my opinion, Tsai discloses Claim 25 for substantially the same reasons as discussed in Ground 1A, Claim 2.

26. Claim 26

“A system according to claim 18, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

176. In my opinion, Tsai discloses Claim 26 for substantially the same reasons as discussed in Ground 1A, Claim 9.

27. Claim 27

a. 27[pre]

“A system for image coding comprising:

177. In my opinion, Tsai discloses 27[pre] for substantially the same reasons as discussed in Ground 1A, Claim 18[pre].

b. 27[a]

“an input for receiving an image as a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

178. In my opinion, Tsai discloses 27[a] for substantially the same reasons as discussed in Ground 1A, Claim 18[a].

c. 27[b]

“a transform coder for performing a transform coding operation on a block of pixels to produce a corresponding block of transform coefficient values;”

179. In my opinion, Tsai discloses 27[b] for substantially the same reasons as discussed in Ground 1A, Claim 18[b].

d. 27[c]

“a scanner for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;”

180. In my opinion, Tsai discloses 27[c] for substantially the same reasons as discussed in Ground 1A, Claim 18[c].

e. 27[d]

“a run-length coder for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

181. In my opinion, Tsai discloses 27[d] for substantially the same reasons as discussed in Ground 1A, Claim 18[d].

f. 27[e]

“a context-based coder for assigning the second numbers to one of a plurality of contexts representative of the second numbers, and operative to assign the second number of a number pair to a context at least partly in dependence on the first number of the number pair.”

182. In my opinion, Tsai discloses 27[e] for substantially the same reasons as discussed in Ground 1A, Claim 18[e].

28. Claim 28

“A system according to claim 27, wherein said context-based coder is a context-based arithmetic coder.”

183. In my opinion, Tsai discloses Claim 28 for substantially the same reasons as discussed in Ground 1A, Claim 11.

29. Claim 29

“A system according to claim 27, wherein said context-based coder is a context-based binary arithmetic coder.”

184. In my opinion, Tsai discloses Claim 29 for substantially the same reasons as discussed in Ground 1A, Claim 12.

30. Claim 30

“A system according to claim 29, wherein the context-based coder is operative to map the second numbers of said number pairs to a set of bins, each of which can be assigned to one of either a first or a second value.”

185. In my opinion, Tsai discloses or 30 for substantially the same reasons as discussed in Ground 1A, Claim 13.

31. Claim 31

“A system according to claim 30, wherein the context-based coder is operative to map a number to a given one of said set of bins by assigning the bin to the first value.”

186. In my opinion, Tsai discloses Claim 31 for substantially the same reasons as discussed in Ground 1A, Claim 14.

32. Claim 34

“A system according to claim 27, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

187. In my opinion, Tsai discloses Claim 34 for substantially the same reasons as discussed in Ground 1A, Claim 9.

33. Claim 35

a. 35[pre]

“A computer program embodied in a computer readable medium comprising:”

188. In my opinion, Tsai discloses 35[pre] for substantially the same reasons as discussed in Ground 1A, 18[pre].

b. 35[a]

“machine-readable program code for dividing an image into a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

189. In my opinion, Tsai discloses 35[a] for substantially the same reasons as discussed in Ground 1A, 18[a].

c. 35[b]

“machine-readable program code for performing a transform coding operation on a block of pixel values to produce a corresponding block of transform coefficient values”

190. In my opinion, Tsai discloses 35[b] for substantially the same reasons as discussed in Ground 1A, 18[b].

d. 35[c]

“machine-readable program code for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;”

191. In my opinion, Tsai discloses 35[c] for substantially the same reasons as discussed in Ground 1A, 18[c].

e. 35[d]

“machine-readable program code for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

192. In my opinion, Tsai discloses 35[d] for substantially the same reasons as discussed in Ground 1A, 18[d].

f. 35[e]

“machine-readable program code for assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of second number pair.”

193. In my opinion, Tsai discloses 35[e] for substantially the same reasons as discussed in Ground 1A, 18[e].

34. Claim 36

a. 36[pre]

“A computer program embodied in a computer readable medium comprising:”

194. In my opinion, Tsai discloses 36[pre] for substantially the same reasons as discussed in Ground 1A, 27[pre].

b. 36[a]

“machine-readable program code for dividing an image into a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

195. In my opinion, Tsai discloses 36[a] for substantially the same reasons

as discussed in Ground 1A, 27[a].

c. 36[b]

“machine-readable program code for performing a transform coding operation on a block of pixel values to produce a corresponding block of transform coefficient values;”

196. In my opinion, Tsai discloses 36[b] for substantially the same reasons as discussed in Ground 1A, 27[b].

d. 36[c]

“machine-readable program code for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;”

197. In my opinion, Tsai discloses 36[c] for substantially the same reasons as discussed in Ground 1A, 27[c].

e. 36[d]

“machine-readable program code for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

198. In my opinion, Tsai discloses 36[d] for substantially the same reasons as discussed in Ground 1A, 27[d].

f. 36[e]

“machine-readable code for assigning the second numbers to one of a plurality of contexts representative of the second numbers such that the second number of a number pair is assigned to a context at least partly in dependence on the first number of the number pair.”

199. In my opinion, Tsai discloses 36[e] for substantially the same reasons

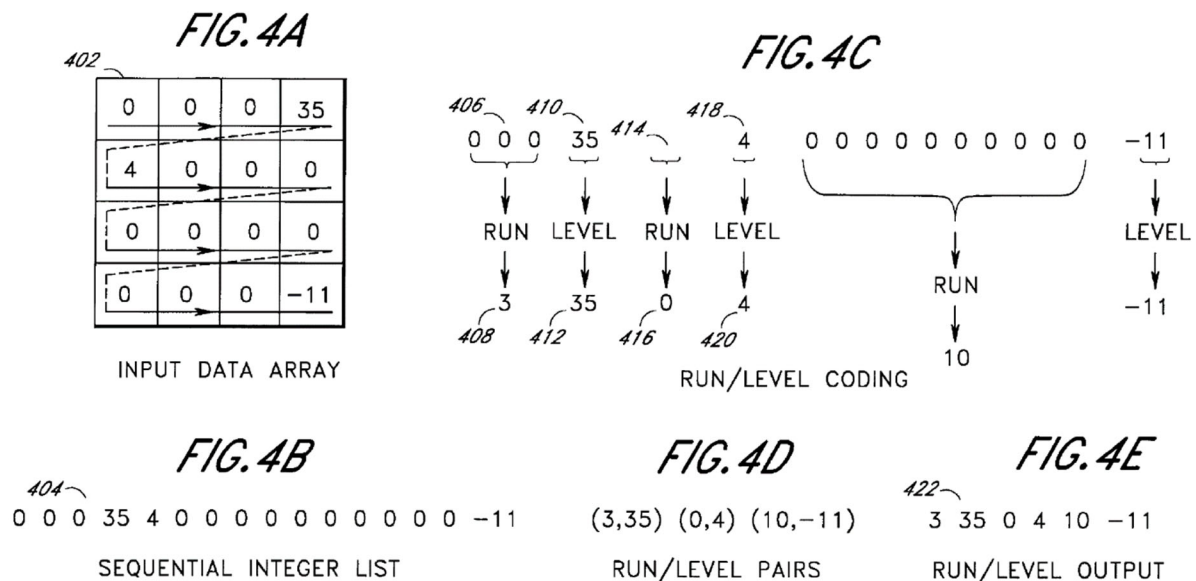
as discussed in Ground 1A, 27[e].

35. Claim 37

a. 37[pre]

“A method of context-based arithmetic encoding in which an array of data symbols is represented with a code-word, the method comprising:”

200. In my opinion, Tsai discloses this limitation. Tsai explains that entropy coder 116 is an adaptive arithmetic coder that performs context-based arithmetic encoding. FIGS. 4A-E of Tsai (reproduced below) show that an array of integer values (*an array of data symbols*) can be represented as a codeword output (e.g., in FIG. 4E) (*a code-word*).



Tsai, FIGS. 4A-E.

b. 37[a]

“representing the data symbols in said array as number pairs comprising a first number and a second number; and”

201. In my opinion, Tsai discloses 37[a] for substantially the same reasons as discussed in Ground 1A, 1[a].

c. 37[b]

“assigning the first number of a first number pair to a context selected from a plurality of contexts representative of the first numbers at least partly in dependence on a first number of a second number pair.”

202. In my opinion, Tsai discloses 37[b] for substantially the same reasons as discussed in Ground 1A, 1[b].

36. Claim 38

a. 38[pre]

“A method of context-based arithmetic encoding in which an array of data symbols is represented with a code-word, the method comprising:”

203. In my opinion, Tsai discloses 38[pre] for substantially the same reasons as discussed in Ground 1A, 37[pre].

b. 38[a]

“representing the data symbols in said array as number pairs comprising a first number and a second number;”

204. In my opinion, Tsai discloses 38[a] for substantially the same reasons as discussed in Ground 1A, 10[a].

c. 38[b]⁴

“assigning the second number of a number pair to a context selected from a plurality of contexts representative of the second numbers at least partly in dependence on the first number of the number pair.”

205. In my opinion, Tsai discloses 38[b] for substantially the same reasons as discussed in Ground 1A, 10[b].

37. Claim 39

a. 39[pre]

“A context-based arithmetic encoder operative to represent an array of data symbols with a code-word and comprising:”

206. In my opinion, Tsai discloses 39[pre] for substantially the same reasons as discussed in Ground 1A, 37[pre].

b. 39[a]

“a run-level coder for representing the data symbols in said array as number pairs comprising a first number and a second number;”

207. In my opinion, Tsai discloses 39[a] for substantially the same reasons as discussed in Ground 1A, 18[d].

c. 39[b]

“a context assigner for assigning the first number of a first number pair to a context selected from a plurality of contexts representative of the first numbers at least partly in dependence

⁴ This claim language reflects a correction made in the June 21, 2005 Certificate of Correction. EX1002, 113.

on a first number of a second number pair.”

208. In my opinion, Tsai discloses 39[b] for substantially the same reasons as discussed in Ground 1A, 18[e].

38. Claim 40

a. 40[pre]

“A context-based arithmetic encoder operative to represent an array of data symbols with a code-word and comprising:”

209. In my opinion, Tsai discloses 40[pre] for substantially the same reasons as discussed in Ground 1A, 37[pre].

b. 40[a]

“a run-level coder for representing the data symbols in said array as number pairs comprising a first number and a second number;”

210. In my opinion, Tsai discloses 40[a] for substantially the same reasons as discussed in Ground 1A, 27[d].

c. 40[b]

“a context assigner for assigning the second number of a number pair to a context selected from a plurality of contexts representative of the second numbers at least partly in dependence on the first number of the number pair.”

211. In my opinion, Tsai discloses 40[b] for substantially the same reasons as discussed in Ground 1A, 27[e].

B. Ground 1B: Obviousness over Tsai, VCEG-L28, and VCEG-L13 for Claims 15-16 and 32-33.

1. VCEG-L13

212. VCEG-L13 is a proposal for modification of H.26L from the Video Coding Experts Group under the ITU. On its face it states that it was generated on January 4, 2001 for discussion during January 9-12, 2001. VCEG-L13, 1.

213. VCEG-L13 states that “the universal variable length code (UVLC) used in the entropy coding method of the current H.26L test model is not optimal with respect to coding efficiency” because the codewords in the UVLC are not designed according to the symbol probabilities. Therefore, VCEG-L13 proposes the adoption of context-based adaptive binary arithmetic coding to address the issue. VCEG-L13, Introduction.

214. VCEG-L13 teaches context models for run/level pairs. VCEG-L13, § 3.3. “A non-binary valued symbol will be decomposed into a sequence of binary decisions.” VCEG-L13, § 4. For example, the level information can be represented by multiple bins, each having a value of “0” or “1.” VCEG-L13, § 3.3.1. Similarly, the run information can also be represented by multiple bins, each having a value of “0” or “1.” VCEG-L13, § 3.3.2.

2. Claim 15

“A method according to claim 13, wherein the first value is 1 and

the second value is 0.”

215. In my opinion, Tsai in view of VCEG-L28 and VCEG-L13 discloses the additional limitations of this claim.

216. As discussed in Ground 1A, 1[b], a POSITA looking to implement Tsai’s symbol mapper would have been motivated to incorporate VCEG-L28’s teaching of switching between two contexts based on the previous level value because doing so would improve Tsai’s coding efficiency.

217. In VCEG-L13, “RUN...[is] coded separately” in bins. VCEG-L13, § 3.3. Table 2 of VCEG-L13 shows that each bin can be assigned a value of 1 (*the first value*) or a value of 0 (*the second value*).

Code_number	Binarization
0	1
1	0 1
2	0 0 1
3	0 0 0 1
4	0 0 0 0 1
5	0 0 0 0 0 1
6	0 0 0 0 0 0 1
.	...
Bin_number	1 2 3 4 5 6 7 . .

(a)

MB_type	Binarization
0	0
1	1 0 0
2	1 0 1
3	1 1 0 0 0
4	1 1 0 0 1
5	1 1 0 1 0
6	1 1 0 1 1
7	1 1 1 0 0
8	1 1 1 0 1
9	1 1 1 1 0
Bin_number	1 2 3 4 5

(b)

Table 2: (a) Binarization used for all symbols except for MB_type, (b) Binarization for MB_type

VCEG-L13, Table 2.

218. A POSITA looking to implement Tsai’s teaching of context-oriented

arithmetic coding would have been motivated to incorporate VCEG-L13's teaching of the CABAC coding techniques. Tsai notes that "[i]t is [] desirable to perform the mapping such that the symbols comprising words appear with unequal probability" to "lead to greater efficiency in the entropy coding." Tsai, 10:42-45. VCEG-L13 teaches that, after converting non-binary symbols into binary representations, "each binary decision is encoded with the binary arithmetic coding engine using the probability model which has been supplied by the context modeling stage." VCEG-L13, 2. "[D]uring encoding the current context model will be updated with the previously encoded symbol. Hence, the model keeps track of the actual statistics and provides a fast adaptation." VCEG-L13, 2. For RUN values, "one model for the first bin and the second model for all remaining bins of the binary sequence related to RUN." VCEG-L13, §3.3.2. A POSITA would have recognized that combining VCEG-L13's adaptive context modeling, which uses models adaptive to the probability statistics of the run values, would have achieved the common goal of improving coding efficiency in Tsai.

219. A POSITA would have had a reasonable expectation of success applying VCEG-L13's teaching to non-binary values in Tsai and the combination would have yielded predictable results. As explained in Tsai, "[a]daptive arithmetic coding is known in the art" and "[a]daptive arithmetic coders which adjust a coding model to an input stream as input data are being coded are known in the art." Tsai,

8:30-39, 15:65-16:1. The CABAC coding algorithm in VCEG-L13 refers to context-based adaptive binary arithmetic coding, consistent with the adaptive arithmetic coding techniques disclosed in Tsai. A POSITA would have understood that mapping of the non-binary run symbols onto a sequence of binary decisions using CABAC is consistent with Tsai's teaching.

3. Claim 16

“A method according to claim 13, wherein the second number of a number pair is assigned to a context at least partly in dependence on the bin to which the second number of the number pair is mapped.”

220. In my opinion, Tsai in view of VCEG-L28 and VCEG-L13 discloses the additional limitations of this claim.

221. VCEG-L13 discloses:

▪ 3.3.2→Context-based Coding of RUN Information¶

For each context ~~ctx~~ *run_level(C)* two separate models are provided for the coding of RUN; one model for the first bin and the second model for all remaining bins of the binary sequence related to RUN.¶

VCEG-L13, 5.

222. That is, the context model used to encode a bin of the run value (*the second number of a number pair is assigned to a context*) is at least partly based on whether the bin is the first bin or remaining bins (*the bin to which the second number of the number pair is mapped*).

4. Claim 32

“A system according to claim 30, wherein the first value is 1 and the second value is 0.”

223. In my opinion, Tsai in view of VCEG-L28 and VCEG-L13 discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 1B, Claim 15.

5. Claim 33

“A system according to claim 30, wherein the context-based coder is operative to assign the second number of a number pair to a context at least partly in dependence on the bin to which the second number of the number pair is mapped.”

224. Tsai in view of VCEG-L28 and VCEG-L13 discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 1B, Claim 16.

C. Ground 2: Obviousness over AAPA and Savatier for Claims 1-9, 18-26, 35, 37, and 39.

1. AAPA

225. As I discuss above in § VI.A, the '701 patent's lengthy Background of the Invention section includes many examples of AAPA. EX1001, 1:17-15:61.

2. Savatier

226. Savatier provides a background description of VLC:

Variable length coding, VLC, consists of determining the general statistics of a signal type to discern the frequency of occurrence of the respective amplitude values within the dynamic range of the signal to be coded, and then

forming a code set of variable length codewords representing a similar dynamic range. Thereafter a correspondence is established between the input fixed length codewords and the variable length codewords such that input codewords occurring with the greatest frequency are assigned variable length codes of shorter bit width. Input codewords occurring most frequently are assigned variable length codewords of lesser bit-width than the input codewords thereby effecting on average the data reduction.

Savatier, 1:15-30.

227. Savatier notes that:

It is not uncommon for signal statistics to change such that a particular VLC may become less than efficient. This may be counteracted by continually monitoring the input signal statistics, and when they change by some predetermined measure, changing the variable length code to a more efficient variable length code.

Savatier, 1:38-43. The change of the VLC, however, can bring signaling overhead:

If a plurality of tables are to be used for encoding a signal, VLC table changes at the encoder must be communicated to the receiver. Some known systems transmit the decoding tables whenever a VLC switch is effected. This of course is impractical if the signal statistics change frequently. Other systems transmit a Table Type codeword along with the encoded data, to inform the receiver that a code switch is effective and which of a plurality of tables to use for decoding. For systems that require frequent VLC changes, and have little signal overhead available, the Table Type codewords may overtax this type of system.

Savatier, 1:49-60.

228. To avoid the additional signal overhead, Savatier discloses an adaptive

VLC system in which the selection of VLC codewords depends on previously encoded data. Savatier, Abstract, 1:5-8, 4:32-34, 2:3-19 (“the VLC table used for each successive codeword is made dependent upon the table utilized for the lastmost codeword and the relative value of the lastmost codeword”). In Savatier, a plurality of tables is respectively preprogrammed with VLC codes associated with anticipated signal statistics. Savatier, Abstract, 2:3-9. A VLC table is adaptively selected for coding each codeword based on the encoded data to account for changes in signal statistics. Savatier, 4:29-34; 1:38-48.

229. FIG. 3 (below) shows an adaptive variable length encoder (“AVLC”) in an embodiment. Savatier, 3:54-55, 4:29-34; *see id.*, 2:27-28. DCT coefficients are quantized and run-length encoded, then output as RUN/LEVEL codewords to VLC 11. Savatier, 3:55-4:4, 5:26-28. VLC Table Logic 15 (red) “determin[es] the VLC table to be used for each respective [RUN/LEVEL] codeword.” Savatier, 4:29-32; *see id.*, 4:4-12.

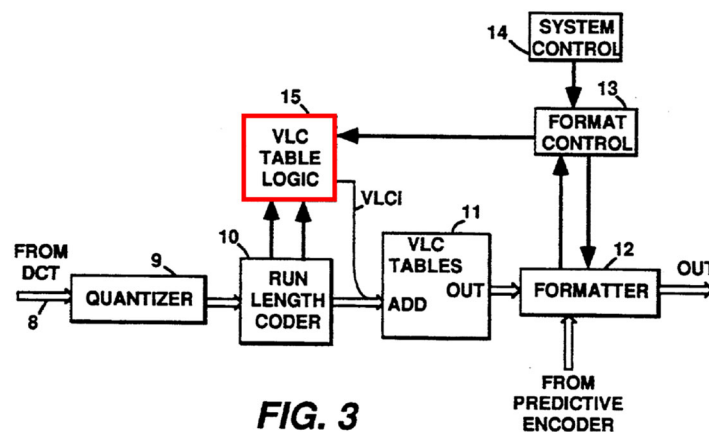


FIG. 3

230. FIG. 4 (below) shows an adaptive variable length decoder (“AVLD”) “arranged to perform the reciprocal operation of the AVLC shown in FIG. 3.” Savatier, 4:35-38, 5:7-12; *see id.*, 2:29-30. The AVLD “performs successive sequences of[:] decoding a codeword, determining its length and shifting the input bit stream by the number of bits corresponding to the length of the codeword decoded before decoding the next codeword.” Savatier, 4:46-52. Table Select Logic 30 “is responsive to the decoded data for determining which of a plurality of decoding tables resident in the memory 28 should be utilized for decoding the next variable length codeword.” Savatier, 5:7-12; *see id.*, 4:37-38.

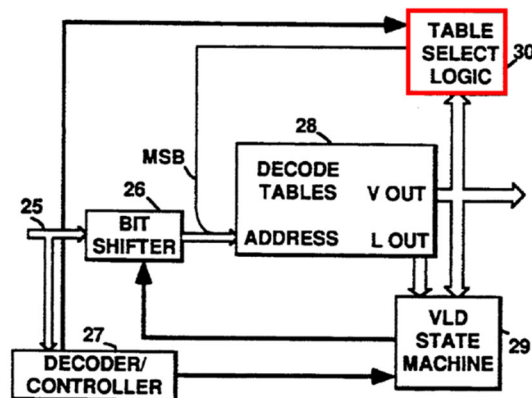


FIG. 4

Savatier, FIG. 4 (annotated).

231. Savatier explains that “the AVLC/AVLD utilize two coding/decoding tables.” Savatier, 5:13-15. “The VLC0 (and reciprocal VLD0) table is optimized for smaller valued coefficients[.]” Savatier, 5:16-17. “[T]he VLC1 (VLD1) table is optimized for larger valued coefficients.” Savatier, 5:17-18.

232. “The adaptivity of the variable length coding/decoding is dependent upon...the information contained in respective [RUN/LEVEL] codewords provided by the run length coder.” Savatier, 5:21-28. For example, “[a]...criterion for selecting coding/decoding tables for each successive codeword is the character of the immediately preceding codeword.” Savatier, 5:31-33.

3. Claim 1

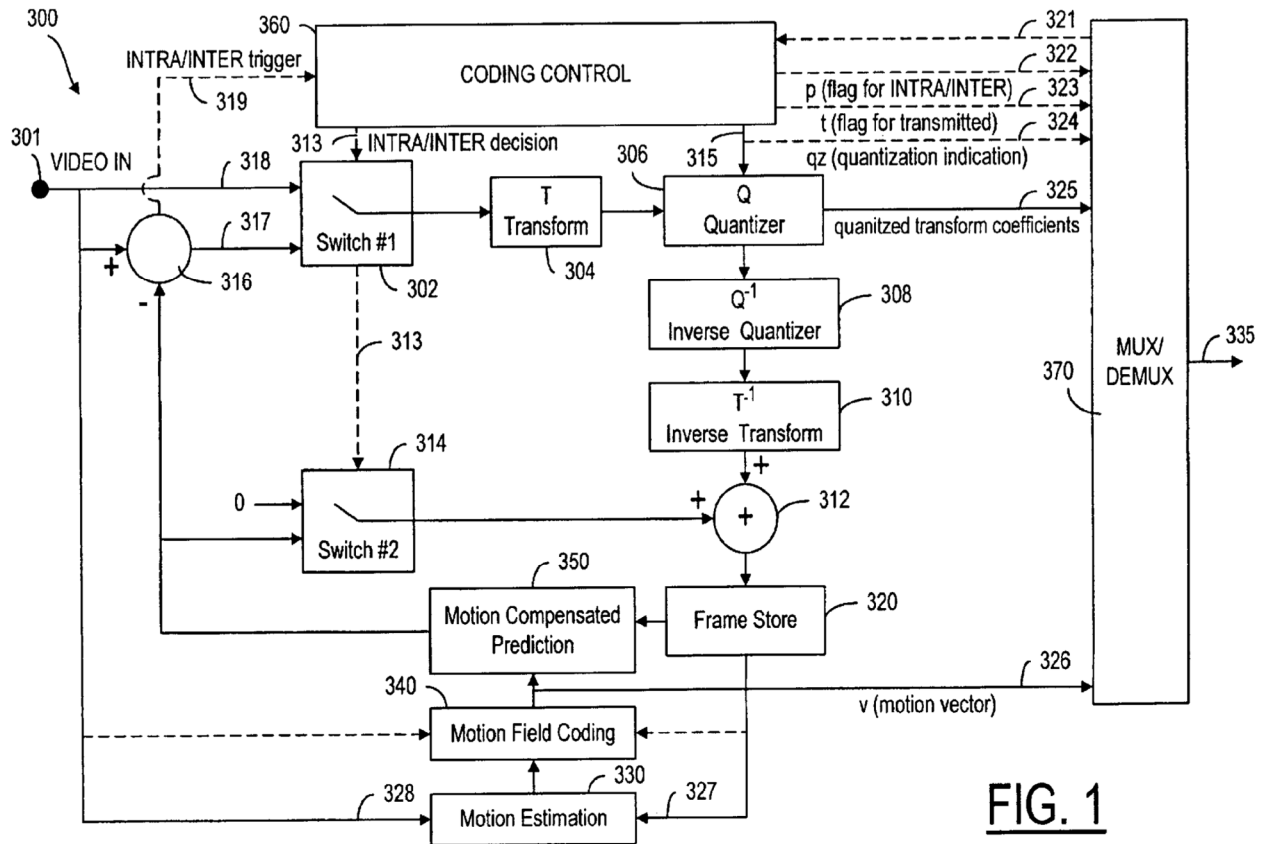
a. 1[pre]

“A method of image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values being scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, the method comprising the steps of:”

233. To the extent this preamble is limiting, AAPA discloses the preamble of claim 1.

234. AAPA discloses a method of image coding within the framework of H.26L Video Coding Standard, which a POSITA would have understood to be *a method of image coding*. EX1001, 11:36-47. The operation of the encoder in FIG. 1 (below) and the associated disclosure provides a comprehensive overview of a generic hybrid video encoder based on H.26L that performs *a method of image coding*. Encoder 300 in FIG. 1 (below) utilizes motion-compensated prediction and block-based transform coding to reduce temporal and spatial redundancy in video

sequences, enabling efficient compression of video frames into a compressed video bit-stream. EX1001, 3:45-11:35.



EX1001, FIG. 1.

235. AAPA discloses that for encoder operations, “[e]ach frame of uncompressed video provided from the video source to input 301 is received and processed macroblock-by-macroblock, preferably in raster-scan order.” EX1001, 4:51-54, FIG. 1. “[E]ach macroblock comprises four 8x8 pixel blocks of luminance values and two spatially corresponding 8x8 pixel blocks of chrominance values,” where “[e]ach pixel, in turn, is represented by a certain number of bits, which carry information about the brightness (luminance) and/or color (chrominance) of the

pixel.” EX1001, 5:35-40, 1:22-24.

236. Therefore, AAPA discloses that a frame of uncompressed video (*an image*) is provided by the encoder macroblock-by-macroblock (*is divided in an encoder into a plurality of blocks*), and each macroblock includes 8x8 pixels, each having luminance and chrominance values (*having a plurality of pixels, each pixel having a pixel value*).

237. AAPA also discloses “encoder 300...comprises a transformation unit 304 [that]...perform[s] a block-based discrete cosine transform (DCT).” EX1001, 4:32-36. As shown in FIG. 1 of AAPA, as the input signal is received, “the blocks of luminance and chrominance values which make up the macroblock are passed to the DCT transformation block 304, which performs a 2-dimensional discrete cosine transform on each block of values, producing a 2-dimensional array of DCT coefficients for each block.” EX1001, 5:30-35. Thus, AAPA discloses a DCT transformation (*a transform coding operation*) performed on a block of luminance and chrominance values (*performed on a block of pixels*) to produce a 2-dimensional array of DCT coefficients (*to produce a corresponding block of transform coefficient values*).

238. AAPA further discloses that “[t]he DCT coefficients for each block are passed to the quantizer 306” and “[t]he quantized DCT coefficients for each block are passed from the quantizer 306 to the video multiplex coder 370.” EX1001, 5:41-

42, 5:56-57. “The video multiplex coder 370 orders the transform coefficients for each block using a zigzag scanning procedure. This operation converts the two-dimensional array of quantized transform coefficients into a one-dimensional array.” EX1001, 5:58-62; *see id.*, 14:46-61, FIGS. 3, 6a-6b (below). AAPA thus discloses that the DCT coefficients are scanned in a zigzag order (*the block of transform coefficient values being scanned in a given scanning order*) to produce a one-dimensional array (*a scanned array of coefficient values arranged according to the scanning order*).

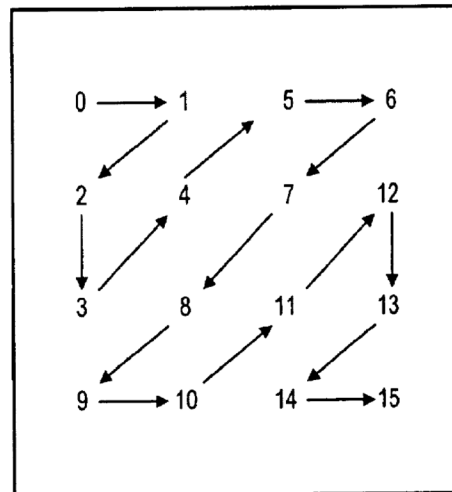


FIG. 3

EX1001, FIG. 3.

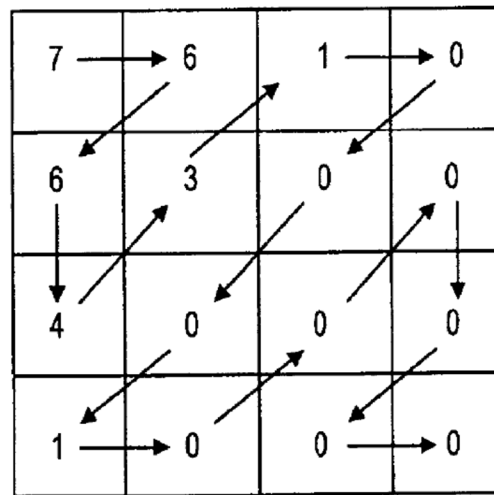


FIG. 6a

07	06	06	04	03	01	31	0
70	60	60	40	30	10	13	0

↑↑ Run
↑ Level

FIG. 6b

FIG. 6b

EX1001, FIGS. 6a-6b.

b. 1[a]

“representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number; and”

239. In my opinion, AAPA discloses this limitation. AAPA discloses that “the video multiplex coder 370 represents each non-zero quantized coefficient in the one dimensional array by two values, referred to as level and run. Level is the value of the quantized coefficient and run is the number of consecutive zero-valued

7	→	6	
	↙		
6		↗	
↓			
4		↗	
	↙		
1	→	0	

EX1001, FIG. 6a.

104

07	06	06	04	03	01	31	0
70	60	60	40	30	10	13	0

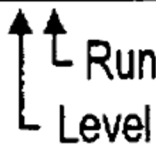


FIG. 6b

EX1001, FIG. 6b.

241. AAPA thus discloses representing the coefficients in the one-dimensional array (*the coefficient values in the scanned array*) using multiple two-value pairs (*a plurality of number pairs*), each pair as level (*a first number*) and run (*a second number*).

c. 1[b]

“assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair.”

242. AAPA in view of Savatier teaches this limitation.

243. APPA discloses the CABAC method in which a context-based binary arithmetic encoder 700 performs a binarization process to map a symbol (e.g., a level or run number) to a “sequence of bins, each of which has a corresponding bin number and can be assigned a value of either 0 or 1.” EX1001, 12:46-49; *see id.*, 14:66-15:3. Using the (6, 0) pair in FIG. 6b and Table 1 (annotated below) as an example,

“6” (red) is the level number , which is binarized to a sequence of bins: 0000001

(red), and “0” (blue) is the run number, which is binarized to bin “1” (blue).

EX1001, 12:46-49, Table 1, FIG. 6b.

FIG. 6b

07	06	06	04	03	01	31	0
70	60	60	40	30	10	13	0

↑ Run
↑ Level

TABLE 1

value	Bin Sequence						
0	1						
1	0	1					
2	0	0	1				
3	0	0	0	1			
4	0	0	0	0	1		
5	0	0	0	0	0	1	
6	0	0	0	0	0	0	1
bin_nr.	1	2	3	4	5	6	7

EX1001, FIG. 6b and TABLE 1 (annotated).

244. Then, each of the bins of the level number is “assigned to a so-called ‘context’ (hence the name context-based arithmetic coding)...[and] each bin assigned to a particular context is assumed to have a similar probability of containing the value 1 or 0 as the other bins belonging to that context.” EX1001, 12:65-13:4; *see*

also id., 15:3-15, FIG. 6c. Again, using the (6, 0) pair in FIG. 6b as an example, the sequence of bins representing the level number—00000010—is assigned to different contexts, as illustrated in annotated FIG. 6c and Table 1 below:

value	Bin Sequence							
0	1							
1	0	1						
2	0	0	1					
3	0	0	0	1				
4	0	0	0	0	1			
5	0	0	0	0	0	1		
6	0	0	0	0	0	0	1	
...	.	.	.	.	.	.	.	.
bin_nr.	1	2	3	4	5	6	7	.

bin number	8	7	6	5	4	3	2	1	0
Context 3	0	0	0	0	0	0	0	0	0
Context 2	0	0	0	0	0	0	0	0	0
Context 1	0	0	0	0	0	0	0	0	0

Level: 7, 0, 6, 0, 4, 0, 3, 0, 1, 0, 1, 3, 0
Run: 0, 6, 0, 4, 0, 3, 0, 1, 0, 1, 3, 0, 0
EOB: 0

FIG. 6c

EX1001, FIG. 6c and TABLE 1 (annotated).

245. The first bin “0” (green) is assigned to Context1 (green), the second bin “0” (yellow) is assigned to Context2 (yellow), and the rest of the bin sequence “000010” (blue) is assigned to Context3 (blue). EX1001, 15:3-15, FIG. 6c, Table 1; *see id.*, 23:3-11 (explaining the four contexts specified in H.26L TML8).

246. Thus, the CABAC scheme of H.26L described in AAPA discloses assigning level values (*the first numbers*) to at least one of Context1, Context2, Context3 representing the level values (*one of a plurality of contexts representative of the first numbers*).

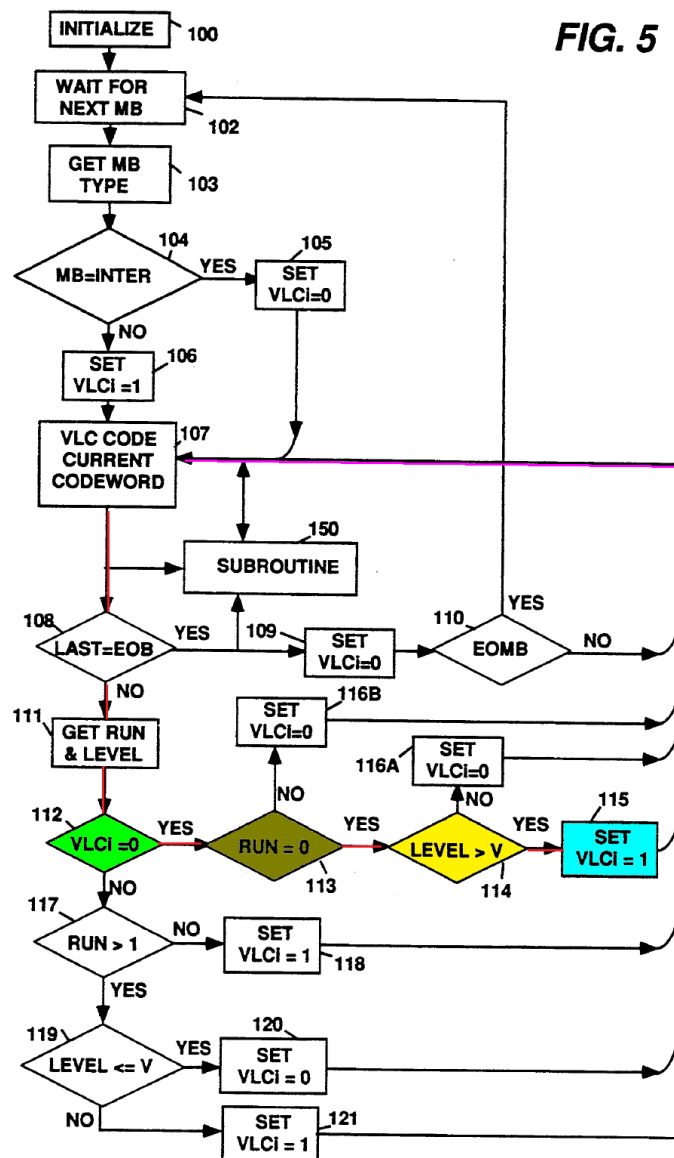
247. Savatier discloses adaptively coding run/level data based on prior data. Savatier, Abstract, 1:5-8, 4:32-34. Savatier provides a plurality of VLC encoding

tables, “respective ones of the tables being preprogrammed with VLC codes associated with anticipated signal statistics.” Savatier, 2:6-9. Here, the grouping of the signal statistics in tables corresponds to the concept of contexts. Specifically, Savatier teaches selecting a VLC coding/decoding table of codes having similar anticipated probabilities (i.e., a “context”) based upon “the table utilized for the lastmost codeword and the relative value of the lastmost codeword.” Savatier, Abstract, 2:3-9.

248. Savatier explains that “[a]...criterion for selecting coding/decoding tables for each successive codeword is the character of the immediately preceding codeword.” Savatier, 5:31-33. “If the coder (decoder) is currently conditioned to use coding/decoding table VLC0, and if the RUN value is zero and the LEVEL value is greater than a predetermined value V (for example V=1), coding/decoding is switched to coding/decoding table VLC1.” Savatier, 5:33-40. On the other hand, “if the coder (decoder) is currently conditioned to use coding/decoding table VLC1, and if RUN is greater than one and the value LEVEL is less than or equal to V, the system switches to coding/decoding table VLC0.” Savatier, 5:40-44. Savatier thus discloses selecting coding/decoding tables for each successive codeword based on the LEVEL value of the immediately preceding codeword.

249. Savatier describes “[t]he operation of the [coding/decoding] TABLE LOGIC” with relation to FIG. 5 (annotated below). Savatier, 2:31-32; *see id.*, 5:62-

6:35. “After a codeword is variable length encoded (decoded) [107; pink], its RUN and LEVEL components [of a codeword] are accessed [111]. Then, if the system is currently conditioned to use the encoding (decoding) table VLC0 [112; green] and the RUN value of the last codeword is equal to zero (for example) [113; brown] and the value of LEVEL is greater than V [114; yellow] the system is switched [115; blue] to utilize table VLC1 to encode/decode [107] the next codeword.” Savatier, 6:15-26.



Savatier, Figure 5.

250. A POSITA would have thus understood that the selection of the VLC table for the current codeword 107 is based on the LEVEL value in previous codeword 114.

251. Savatier's plurality of tables are respectively preprogrammed with VLC codes associated with anticipated signal statistics (i.e., "contexts"). Savatier's table

selection therefore constitutes context selection/assignment. Savatier thus teaches that table selection (*context assignment*) for a current codeword including LEVEL (*the first number of a first number pair*) is based on the LEVEL of a previous codeword (*a first number of a second number pair*).

252. A POSITA would have implemented Savatier's adaptive table selection method in AAPA's H.26L system to improve coding efficiency. CABAC is an adaptive method that takes into account changes in the statistical characteristics of the information being coded. EX1001, 12:27-34. In CABAC, probability estimates for each assigned context are updated after the encoding of previous bins. EX1001, 15:16-18. Savatier explains that "the switching of encode/decode tables as a function of RUN and LEVEL variables tends to enhance overall system coding efficiency." Savatier, 8:9-11. For example, "[i]f...the LEVEL proves to be small, it can be anticipated that the next codeword may have similar attributes." Savatier, 8:13-15. "Hence the system can be switched to take advantage of the similar attributes." Savatier, 8:15-16. A POSITA would have understood that implementing Savatier's teachings in AAPA's H.26L system by referencing the previous LEVEL value would realize Savatier's stated benefits in that system.

253. AAPA's binarization process maps levels and runs "onto a sequence of bins[,] and each bin is assigned to a context based on its bin number." EX1001, 14:40-45. A POSITA would have been motivated to further consider the level in the

preceding run-level pair (as taught in Savatier) for context assignment for each bin of the current level number because, by anticipating the characteristics of subsequent codewords and adapting the context assignments accordingly, the modified H.26L system would achieve better data compression and higher efficiency. Savatier, 8:9-26. For example, in the combined system, using the two pairs (7, 0) and (6, 0) annotated in red in AAPA's Figure 6b below, the current level number 6 is binarized to 00000010 (according to Table 1 of AAPA), and the preceding level number is 7. Per Savatier's teachings, the context for each bin of 6 would be assigned based on the preceding bin values of 7.

07	06	06	04	03	01	31	0
70	60	60	40	30	10	13	0

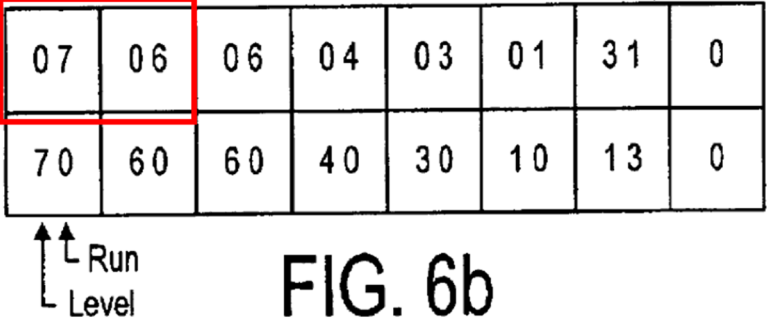


FIG. 6b

EX1001, FIG. 6b (annotated).

TABLE 1

value	Bin Sequence							
0	1							
1	0	1						
2	0	0	1					
3	0	0	0	1				
4	0	0	0	0	1			
5	0	0	0	0	0	1		
6	0	0	0	0	0	0	1	
...	.	.	.	.	.	.	.	.
bin_nr.	1	2	3	4	5	6	7	.

EX1001, Table 1 (annotated).

254. In the proposed combination, the first bin of the current level number 6 ('0') would be assigned to a context (e.g., VLC0 under CABAC) based on the bin number (i.e., 1) and/or the preceding level number 7. Similarly, the second to seventh bin of the current level number 6 would be assigned to contexts based on the bin number (i.e., 2...7) and/or the preceding level number 7. Savatier, 6:15-35.

255. For several reasons, a POSITA would have found it obvious to incorporate Savatier's method of adaptively selecting a VLC table based on previously encoded data into the generic hybrid video encoder/decoder disclosed in AAPA.

256. First, a POSITA would have recognized that Savatier and AAPA are in the same field of invention because both are directed to MPEG coding systems, specifically the entropy coding (VLC) stage of the coding process aiming to improve coding efficiency. Savatier, 1:64-68, 2:39-48, FIGS. 1-2; EX1001, 11:35-12:34.

Thus, a POSITA would have found it obvious that AAPA's system would have benefited from Savatier's complementary coding techniques that had the same or similar goal for improving coding efficiency.

257. Second, a POSITA would have recognized that Savatier would improve the AAPA H.26L system. Savatier explains "the switching of encode/decode tables as a function of RUN and LEVEL variables tends to enhance overall system coding efficiency." Savatier, 8:9-11. "If...the LEVEL proves to be small, it can be anticipated that the next codeword may have similar attributes." Savatier, 8:13-15. "Hence the system can be switched to take advantage of the similar attributes." Savatier, 8:15-16. The CABAC in H.26L uses adaptive probability estimation to encode binarized values based on their contexts. EX1001, 13:20-28. A POSITA would have recognized the benefits of Savatier's approach to be equally applicable to AAPA's H.26L because, by anticipating the characteristics of subsequent codewords and adapting the context assignments accordingly, the modified H.26L system would achieve better data compression with higher efficiency.

258. Further, a POSITA would have reasonably expected success in adding Savatier's method into the CABAC of H.26L. According to AAPA, the H.26L standard already includes a CABAC method that assigns contexts to run and level values based on factors like block type and bin number. Savatier's table selection is also based on the macroblock (MB) type. Savatier, 5:66-6:5. In other words, the

H.26L framework is already structured to accommodate a statistically adaptive coding scheme like Savatier's. A POSITA would have recognized that the combination uses a known technique (Savatier's referencing a prior codeword) to a similar known device ready for improvement (AAPA's generic hybrid video encoder/decoder) to yield predictable results (an encoder/decoder having enhanced overall system coding efficiency).

4. Claim 2

“A method according to claim 1, wherein the first number of the first number pair is assigned to a context at least partly in dependence on the first number of a preceding number pair in the scanning order.”

259. AAPA in view of Savatier teaches Claim 2 for substantially the same reasons as discussed in Ground 2, 1[b]. Specifically, Savatier teaches table selection (i.e., context assignment) for a given codeword based on the level value of the preceding run/level value pair (*the first number of a preceding number pair in the scanning order*).

5. Claim 3

“A method according to claim 1, wherein said plurality of contexts are contexts of a context-based arithmetic coder.”

260. AAPA discloses the additional limitations of this claim because AAPA's CABAC encoder 700 uses CABAC coding scheme for entropy coding (*a context-based arithmetic coder*). EX1001, 12:41-45, 13:20-28, FIG. 4; *see id.*,

11:63-67, 12:65-13:4, 15:3-15, FIG. 6c.

6. Claim 4

“A method according to claim 1, wherein said plurality of contexts are contexts of a context-based binary arithmetic coder.”

261. AAPA discloses the additional limitations of this claim because AAPA’s CABAC encoder 700 uses CABAC coding scheme for entropy coding (*a context-based binary arithmetic coder*). EX1001, 12:41-45, 13:20-28, FIG. 4; *see id.*, 11:63-67, 12:65-13:4, 15:3-15, FIG. 6c.

7. Claim 5

“A method according to claim 4, further comprising the step of: mapping the first numbers of said number pairs to a set of bins, each of which can be assigned one of either a first or a second value.”

262. In my opinion, AAPA discloses the additional limitations of this claim. AAPA describes the CABAC arithmetic coding method as one where non-binary values are first converted to binary values in a process known as binarization mapping a symbol (e.g., a level or run number) to a sequence of **bins**, each of which can be assigned a value of either 1 (first value) or 0 (second value). EX1001, 12:42-50, Table 1 (annotated below for the bin sequence “0001” for value “3”).

TABLE 1

value	Bin Sequence							
0	1							
1	0	1						
2	0	0	1					
3	0	0	0	1				
4	0	0	0	0	1			
5	0	0	0	0	0	1		
6	0	0	0	0	0	0	1	
...	.	.	.	.	.	.	.	.
bin_nr.	1	2	3	4	5	6	7	.

EX1001, TABLE 1(annotated).

8. Claim 6

“A method according to claim 5, wherein mapping of a number to a given one of said set of bins is indicated by assigning the bin to the first value.”

263. In my opinion, AAPA discloses the additional limitations of this claim. For example, in the CABAC described in AAPA, value “0” is mapped to one bin (‘1’). EX1001, 12:42-50, Table 1 (annotated below for the bin ‘1’ for value “0”). Thus, AAPA discloses mapping the level number 0 (*mapping of a number*) to a bin (*to a given one of said set of bins*) that is assigned to the binary value of 1 (*assigning the bin to the first value*).

TABLE 1

value	Bin Sequence							
0	1							
1	0	1						
2	0	0	1					
3	0	0	0	1				
4	0	0	0	0	1			
5	0	0	0	0	0	1		
6	0	0	0	0	0	0	1	
...	.	.	.	.	.	.	.	.
bin_nr.	1	2	3	4	5	6	7	.

EX1001, TABLE 1(Annotated).

9. Claim 7

“A method according to claim 5, wherein the first value is 1 and the second value is 0.”

264. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 5. As shown in Table 1 of the '701 patent (below), the value of a level (e.g., 3) is represented using bins having value of 1 (*the first value is 1*) or value of 0 (*the second value is 0*). EX1001, 12:42-50.

TABLE 1

value	Bin Sequence								
0	1								
1	0	1							
2	0	0	1						
3	0	0	0	1					
4	0	0	0	0	1				
5	0	0	0	0	0	1			
6	0	0	0	0	0	0	1		
...	.	.	.	.	.	.	.	.	.
bin_nr.	1	2	3	4	5	6	7	.	.

EX1001, TABLE 1 (annotated).

10. Claim 8

“A method according to claim 5, wherein the first number of a number pairs is assigned to a context at least partly in dependence on the bin to which the first number of said number pair is mapped.”

265. In my opinion, AAPA discloses the additional limitations of this claim.

266. In AAPA, the assignment of the context for a bin is at least partly based on the position of the bin in the multiple bins. For example, the first bin “0” in the **bin sequence** for value 3 is assigned to context 1 (in green), the second bin “0” is assigned to context 2 (in yellow), and the third bin “0” and fourth bin “1” are both assigned to context 3 (in blue). EX1001, 15:3-15, FIG. 6c, Table 1 (annotated below); *see id.*, 23:3-11 (explaining the four contexts specified in H.26L TML8).

Thus, the assignment of the context, e.g., Context1, Context 2, or Context 3, for a bin representing the level value (*the first number of a number pairs is assigned to a context*) based on the location of the bin in the bin sequence (*at least partly in dependence on the bin to which the first number of the said number pair is mapped*).

TABLE 1

value	Bin Sequence									
0	1									
1	0	1								
2	0	0	1							
3	0	0	0	1						
4	0	0	0	0	1					
5	0	0	0	0	0	1				
6	0	0	0	0	0	0	1			
...	.	.	.	.	.	.	.			
bin_nr.	1	2	3	4	5	6	7			

bin number	8	7	6	5	4	3	2	1	0	EOB
Context 3	1	0	0	0	0	0	0	0	0	0
	0	0	1	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	1
Context 2	0	0	0	0	0	0	0	0	1	0
Context 1	0	1	0	1	0	1	0	1	0	0

7 0
6 0 6 0 4 0
3 0 1 0 1 3
0
Level
Run
EOB

FIG. 6c

EX1001, FIG. 6c (Annotated), TABLE 1(Annotated).

11. Claim 9

“A method according to claim 1, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

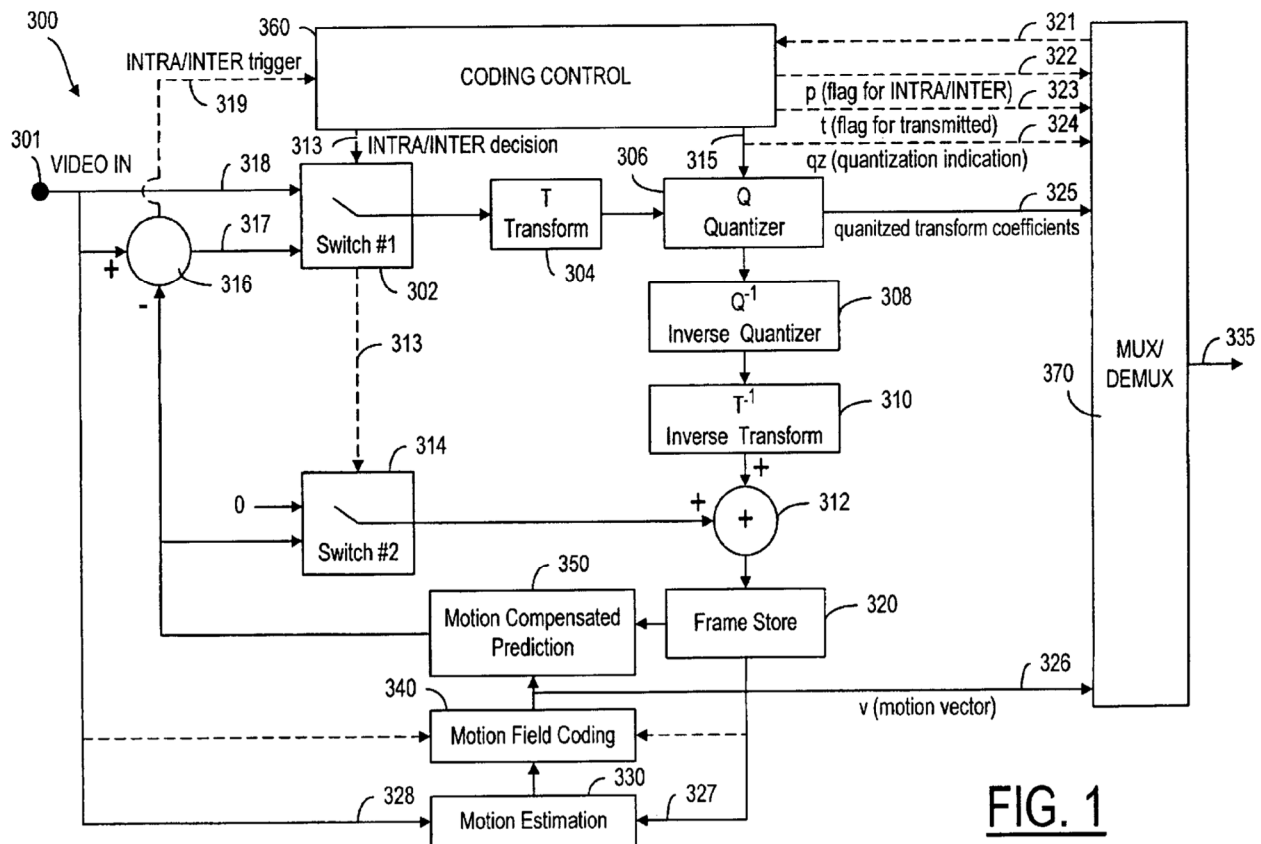
267. In my opinion, AAPA discloses the additional limitations of this claim.

Specifically, AAPA discloses that the level value (*said first number*) is “the value of the quantized coefficient” (*indicative of a non-zero coefficient value*) and that the run (*said second number*) is “the number of consecutive zero-valued coefficients preceding the coefficient in question” (*indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value*). EX1001, 6:9-12.

12. Claim 18**a. 18[pre]****“A system for image coding comprising:”**

268. If this preamble is limiting, AAPA discloses a system for image coding.

FIG. 1 of AAPA (reproduced below) illustrates a prior art “video encoder that employs a combination of INTRA- and INTER-coding to produce a compressed (encoded) video bit-stream.” EX1001, 4:27-30, FIG. 1; *see also* Ground 2, 1[pre].



EX1001, FIG. 1.

b. 18[a]

“an input for receiving an image as a plurality of blocks having a

plurality of pixels, each pixel having a pixel value;”

269. AAPA teaches this limitation because its encoder 300 comprises an input 301 “for receiving a digital video signal from a camera or other video source” EX1001, 4:32-34. “Each frame of uncompressed video provided from the video source to input 301 is received and processed macroblock-by-macroblock, preferably in raster-scan order.” EX1001, 4:51-54. “[E]ach macroblock comprises four 8x8 pixel blocks of luminance values and two spatially corresponding 8x8 pixel blocks of chrominance values,” where “[e]ach pixel, in turn, is represented by a certain number of bits, which carry information about the brightness (luminance) and/or color (chrominance) of the pixel.” EX1001, 1:22-24, 5:35-40.

270. Therefore, AAPA discloses an input 301 (*input*) for receiving a frame of uncompressed video (*an image*) that is to be processed macroblock-by-macroblock (*as a plurality of blocks*), and each macroblock includes 8x8 pixels, each having luminance and chrominance values (*having a plurality of pixels, each pixel having a pixel value*).

c. 18[b]**“a transform coder for performing a transform coding operation on a block of pixels to produce a corresponding block of transform coefficient values;”**

271. AAPA discloses this limitation. AAPA discloses a transformation unit 304 (*a transform coder*) that performs “a 2-dimensional discrete cosine transform

on each block of values, producing a 2-dimensional array of DCT coefficients for each block.” EX1001, 5:32-40; *see* Ground 2, 1[pre]. “[T]he blocks of luminance and chrominance values which make up the macroblock are passed to the DCT transformation block 304, which performs a 2-dimensional discrete cosine transform on each block of values, producing a 2-dimensional array of DCT coefficients for each block.” EX1001, 5:30-35.

272. Thus, AAPA discloses a transformation unit 304 (*a transform coder*) for performing a DCI transform (*for performing a transform coding operation*) on a block of luminance and chrominance values (*on a block of pixels*) to produce a 2-dimensionanl array of DCT coefficients (*to produce a corresponding block of transform coefficient values*).

d. 18[c]

“a scanner for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values, arranged according to the scanning order;”

273. AAPA teaches this limitation. AAPA discloses a video multiplex coder 370 “orders the transform coefficients for each block using a zigzag scanning procedure,” which “converts the two-dimensional array of quantized transform coefficients into a one-dimensional array.” EX1001, 5:56-64, FIG. 3; Ground 2, 1[pre]. AAPA thus discloses a video multiplex coder 370 (*a scanner*) that scans transform coefficients for each block using a zigzag scanning procedure (*for*

scanning the block of transform coefficient values in a given scanning order) to produce a one-dimensional array (to produce a scanned array of coefficient values, arranged according to the scanning order).

e. 18[d]

“a run-level coder for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

274. AAPA discloses this limitation. AAPA does not explicitly mention a run-level coder, but it discloses that the video multiplex coder 370 also performs the function to “represent[] each non-zero quantized coefficient in the one dimensional array by two values, referred to as level and run.” EX1001, 6:7-17; 1[a]. AAPA thus discloses the video multiplex coder 370 (*a run-level coder*) that represents each non-zero quantized coefficient in the one dimensional array (*for representing the coefficient values in the scanned array*) by two values (*by a plurality of number pairs*), referred to as level and run (*said number pairs having a first number and a second number*).

f. 18[e]

“a context-based coder for assigning the first numbers to one of a plurality of contexts representative of the first numbers and operative to assign the first number of a first number pair to a context at least partly in dependence on a first number of a second number pair.”

275. AAPA in view of Savatier discloses this limitation. AAPA discloses a

context-based binary arithmetic encoder 700 (specifically, context assignment block 720 thereof) assigns a symbol (level or run number) to a “corresponding context.”

EX1001, 13:20-23, FIG. 4; *see also id.*, 12:65-13:4.

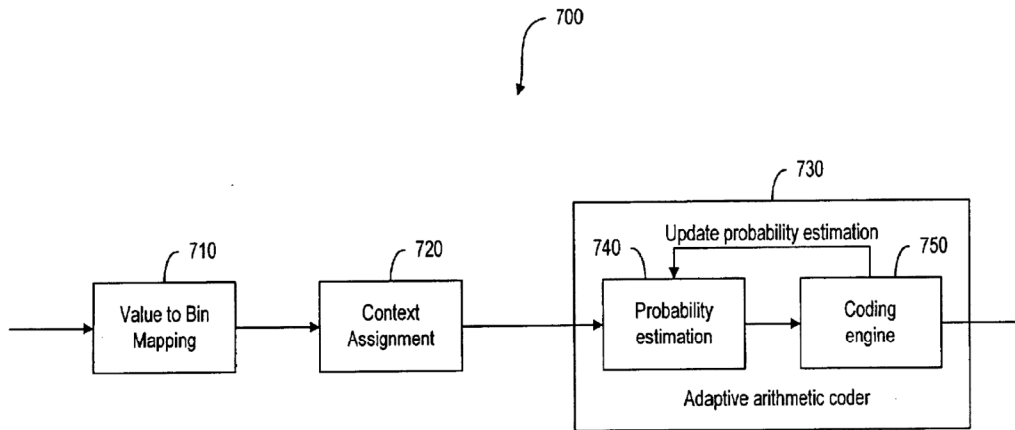


FIG. 4
(PRIOR ART)

EX1001, FIG. 4.

276. Specifically, FIG. 6c of AAPA illustrates the “assignment of bins [(in red blocks)] [converted from a level or run number] to contexts [1(in green), 2 (in yellow), and 3 (in blue)] according to H.26L” ‘701 patent, 15:5-15, Table 1, FIG. 6c (annotated below); *see* Ground 2, 1[b].

value	Bin Sequence							
0	1							
1	0	1						
2	0	0	1					
3	0	0	0	1				
4	0	0	0	0	1			
5	0	0	0	0	0	1		
6	0	0	0	0	0	0	1	
bin_nr.	1	2	3	4	5	6	7	.

	bin number	8	7	6	5	4	3	2	1	0
Context 3	8	1	0	0	0	0	0	0	0	0
	7	0	0	1	0	1	0	0	0	0
	6	0	0	0	0	0	0	0	0	0
	5	0	0	0	0	0	0	1	0	0
	4	0	0	0	0	0	0	0	1	0
	3	0	0	0	0	0	0	0	0	1
Context 2	2	0	0	0	0	0	0	0	0	1
Context 1	1	0	1	0	1	0	1	0	1	0

Level: 7
 Run: 6

EOB

EX1001, FIG. 6c and TABLE 1(Annotated).

278. Further, a POSITA would have understood that encoder 700 is a part of the coder 370 because coder 370 is responsible for entropy coding in system 300, and CABAC is an integral part of this process. EX1001, 7:50-60. As I explain *supra* § V.D, the CABAC algorithm achieves high compression efficiency by assigning different contexts based on probability models, reinforcing the role of encoder 700 in this scheme. This further aligns with Savatier's coding methodology, which emphasizes context-based encoding to optimize compression.

126

(first) number to a context. A POSITA would have found it obvious to add Savatier's coding scheme with a reasonable expectation of success for the same reasons discussed for Ground 2, 1[b]. The combination would yield an efficient encoding system that leverages the strengths of both AAPA's structure and Savatier's detailed context-based coding scheme, with a reasonable expectation of success for improving coding efficiency.

13. Claim 19

“A system according to claim 18, wherein said context-based coder is a context-based arithmetic coder.”

280. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 3.

14. Claim 20

“A system according to claim 18, wherein said context-based coder is a context-based binary arithmetic coder.”

281. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 4.

15. Claim 21

“A system according to claim 20, wherein the context-based coder is operative to map the first numbers of said number pairs to a set of bins, each of which can be assigned to one of either a first or a second value.”

282. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 5.

16. Claim 22

“A system according to claim 21, wherein the context-based coder is operative to map a number to a given one of said set of bins by assigning the bin to the first value.”

283. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 6.

17. Claim 23

“A system according to claim 21, wherein the first value is 1 and the second value is 0.”

284. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 7.

18. Claim 24

“A system according to claim 21, wherein the context-based coder is operative to assign the first number of a number pair to a context at least partly in dependence on the bin to which the first number of said number pair is mapped.”

285. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 8.

19. Claim 25

“A system according to claim 18, wherein the context-based coder is operative to assign the first number of the first number pair to a context at least partly in dependence on the first number of a preceding number pair in the scanning order.”

286. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 2.

20. Claim 26

“A system according to claim 18, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

287. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 2, Claim 9.

21. Claim 35

a. 35[pre]

“A computer program embodied in a computer readable medium comprising:”

288. In my opinion, AAPA discloses 35[pre] for substantially the same reasons as discussed in Ground 2, 18[pre].

b. 35[a]

“machine-readable program code for dividing an image into a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

289. In my opinion, AAPA discloses 35[a] for substantially the same reasons as discussed in Ground 2, 18[a].

c. 35[b]

“machine-readable program code for performing a transform coding operation on a block of pixel values to produce a corresponding block of transform coefficient values”

290. In my opinion, AAPA discloses 35[b] for substantially the same reasons as discussed in Ground 2, 18[b].

d. 35[c]

“machine-readable program code for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;”

291. In my opinion, AAPA discloses 35[c] for substantially the same reasons as discussed in Ground 2, 18[c].

e. 35[d]

“machine-readable program code for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

292. In my opinion, AAPA discloses 35[d] for substantially the same reasons as discussed in Ground 2, 18[d].

f. 35[e]

“machine-readable program code for assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of second number pair.”

293. In my opinion, AAPA in view of Savatier discloses 35[e] for substantially the same reasons as discussed in Ground 2, 18[e].

22. Claim 37

a. 37[pre]

“A method of context-based arithmetic encoding in which an array of data symbols is represented with a code-word, the method

comprising:”

294. In my opinion, AAPA discloses 37[pre] because it describes a method of context-based arithmetic encoding in which a two-dimensional arrangement of quantized DCT coefficient values (*array of data symbols*) is represented with a one-dimensional arrangement of values (e.g., indicated in Figure 6b), such as “7 0” (*a code-word*). EX1001, 14:46-53, FIG. 6b (below).

07	06	06	04	03	01	31	0
70	60	60	40	30	10	13	0

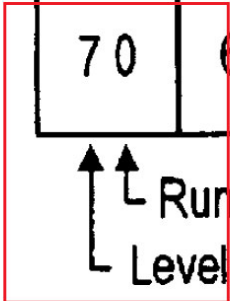


FIG. 6b

EX1001, FIG. 6B (annotated).

b. 37[a]

“representing the data symbols in said array as number pairs comprising a first number and a second number; and”

295. In my opinion, AAPA discloses 37[a] for substantially the same reasons as discussed for Ground 2, 1[a].

c. 37[b]

“assigning the first number of a first number pair to a context

selected from a plurality of contexts representative of the first numbers at least partly in dependence on a first number of a second number pair.”

296. In my opinion, AAPA in view of Savatier discloses 37[b] for substantially the same reasons as discussed for Ground 2, 1[b].

23. Claim 39

a. 39[pre]

“A context-based arithmetic encoder operative to represent an array of data symbols with a code-word and comprising:”

297. In my opinion, AAPA discloses a video multiplex coder 370 operative to represent a two-dimensional arrangement of quantized DCT coefficient values (*array of data symbols*) with a one-dimensional arrangement of values (e.g., indicated in Figure 6b), such as “7 0” (*a code-word*). EX1001, 5:56-64, FIG. 3, 14:46-53, FIG. 6b (below).

07	06	06	04	03	01	31	0
70	60	60	40	30	10	13	0

↑ Run
↑ Level

FIG. 6b

EX1001, FIG. 6b (annotated)

b. 39[a]

“a run-level coder for representing the data symbols in said array as number pairs comprising a first number and a second number;”

298. In my opinion, AAPA discloses 39[a] for substantially the same reasons as discussed in Ground 2, 18[d].

c. 39[b]

“a context assigner for assigning the first number of a first number pair to a context selected from a plurality of contexts representative of the first numbers at least partly in dependence on a first number of a second number pair.”

299. In my opinion, AAPA in view of Savatier discloses 39[b] for substantially the same reasons as discussed in Ground 2, 18[e].

D. Ground 3: Obviousness over AAPA and VCEG-L13 for Claims 10-17, 27-34, 36, 38, and 40.

1. Claim 10

a. 10[pre]

“A method of image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values is scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, the method comprising the steps of:”

300. To the extent this preamble is limiting, AAPA discloses it for substantially the same reasons as discussed in Ground 2, 1[pre].

b. 10[a]

“representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number; and”

301. In my opinion, AAPA discloses this limitation for substantially the same reasons as discussed in Ground 2, 1[a].

c. 10[b]

“assigning the second numbers to one of a plurality of contexts representative of the second numbers such that the second number of a number pair is assigned to a context at least partly in dependence on the first number of the number pair.”

302. VCEG-L13 discloses this limitation, and a POSITA would have found it obvious to add VCEG-L13’s features into the CABAC of H.26L described in AAPA.

303. In VCEG-L13, the level value is first processed and assigned to context(s). VCEG-L13, §3.3.1 (“For a given block C, the LEVEL information is first separated into sign and magnitude”). VCEG-L13 also teaches that the run value of a run-level pair (*the second number of a number pair*) is coded only when the level value of the same run-level pair (*the first number of the number pair*) is not equal to 0. VCEG-L13, §3.3.1 (“If LEVEL≠0 (EOB), the corresponding RUN is coded in a subsequent coding routine.”). In other words, assigning a context model to the run value in a run-level pair (*the second number of a number pair is assigned to a context*) is selectively performed based on the whether the level value in the

same run-level pair is equal to 0 (*at least partly in dependence on the first number of the number pair*).

304. A POSITA would have been motivated to add VCEG-L13's scheme into H.26L (described in AAPA) for several reasons.

305. First, VCEG-L13 proposed "a number of techniques for rapid adaptation to the changing statistics of all syntax elements" for H.26L (described in AAPA). VCEG-L13, Title ("Adaptive Codes for H.26L"), §1. A POSITA would have found it obvious that H.26L would have benefited VCEG-L13's technique by improving coding efficiency while maintaining video quality—when the level value is 0, coding of the corresponding run value is omitted to reduce processing steps.

306. Second, according to VCEG-L13, "[i]f LEVEL≠0 (EOB), the corresponding RUN is coded in a subsequent coding routine," which indicates that the run value is only coded if the level value is non-zero. VCEG-L13, §3.3.1. By introducing this conditional step, VCEG-L13—unlike H.26L described in the AAPA—avoids the computational overhead associated with unnecessarily coding run values. A POSITA would therefore have been motivated to add VCEG-L13's approach into H.26L because the modified H.26L would have allowed fewer operations, saving both processing time and power when coding run values.

307. Third, the AAPA-VCEG-L13 combination would yield an improved system. VCEG-L13 provides empirical evidence of bitrate reductions achieved

through its proposed method. For example, “bitrate savings are in the range of 4.5-15% for coding of a whole sequence and 3.5-17% for pure intra coding” for QCIF sequences, and “5-32% for coding a whole sequence and 4.5-28% for pure intra coding” for CIF-formatted material. VCEG-L13, §6. A POSITA would have recognized that incorporating VCEG-L13’s method into H.26L would enhance compression efficiency across various bit rates. VCEG-L13, §7.

308. A POSITA would have reasonably expected success in adding VCEG-L13’s scheme at least because it is designed to integrate with H.26L as a refinement aimed at improving the test model of H.26L. VCEG-L13, Title, §7 (“We presented a new entropy coding scheme for H.26L based on adaptive, context-based arithmetic coding.”).

2. Claim 11

“A method according to claim 10, wherein said plurality of contexts are contexts of a context-based arithmetic coder.”

309. In my opinion, AAPA discloses the additional limitations of Claim 11 for substantially the same reasons as discussed in Ground 2, Claim 3.

3. Claim 12

“A method according to claim 10, wherein said plurality of contexts are contexts of a context-based binary arithmetic coder.”

310. In my opinion, AAPA discloses the additional limitations of Claim 12 for substantially the same reasons as discussed in Ground 2, Claim 4.

4. Claim 13

“A method according to claim 12, further comprising the step of: mapping the second numbers of said number pairs to a set of bins, each of which can be assigned one of either a first value or a second value.”

311. In my opinion, AAPA in view of VCEG-L13 teaches the additional limitations of Claim 13. As discussed for Ground 2, Claim 5, AAPA teaches mapping of a symbol to a sequence of bins. VCEG-L13 teaches separate coding of RUN and LEVEL. VCEG-L13, §3.3. For coding the RUN information, “one model for the first bin and the second model for all remaining bins of the binary sequence related to RUN.” VCEG-L13, §3.3.2. That is, AAPA in view of VCEG-L13 teaches mapping of RUN information to bins (*mapping the second numbers of said number pairs to a set of bins*), each of which can be assigned a value of either 1 (*a first value*) or 0 (*a second value*).

5. Claim 14

“A method according to claim 13, wherein mapping of a number to a given one of said set of bins is indicated by assigning the bin to the first value.”

312. In my opinion, AAPA discloses the additional limitations of Claim 14 for substantially the same reasons as discussed in Ground 2, Claim 6.

6. Claim 15

“A method according to claim 13, wherein the first value is 1 and the second value is 0.”

313. In my opinion, AAPA discloses the additional limitations of Claim 15

for substantially the same reasons as discussed in Ground 2, Claim 7.

7. Claim 16

“A method according to claim 13, wherein the second number of a number pair is assigned to a context at least partly in dependence on the bin to which the second number of the number pair is mapped.”

314. In my opinion, AAPA in view of VCEG-L13 teaches the additional limitations of Claim 16. As discussed for Ground 2, Claim 8, AAPA teaches that the assignment of the context for a bin is at least partly based on the position of the bin in the multiple bins. VCEG-L13 teaches separate coding of RUN and LEVEL. VCEG-L13, §3.3. For coding the RUN information, “one model for the first bin and the second model for all remaining bins of the binary sequence related to RUN.” VCEG-L13, §3.3.2. That is, AAPA in view of VCEG-L13 teaches that context model assignment for the RUN value (*the second number of a number pair is assigned to a context*) is based on whether the bin is a first bin or one of the remaining bins (*at least partly in dependence on the bin to which the second number of the number pair is mapped*).

8. Claim 17

“A method according to claim 10, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

315. In my opinion, AAPA discloses the additional limitations of Claim 16

for substantially the same reasons as discussed in Ground 2, Claim 9.

9. Claim 27

a. 27[pre]

“A system for image coding comprising:”

316. In my opinion, AAPA discloses 27[pre] for substantially the same reasons as discussed in Ground 2, 18[pre].

b. 27[a]

“an input for receiving an image as a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

317. In my opinion, AAPA discloses 27[a] for substantially the same reasons as discussed in Ground 2, 18[a].

c. 27[b]

“a transform coder for performing a transform coding operation on a block of pixels to produce a corresponding block of transform coefficient values;”

318. In my opinion, AAPA discloses 27[b] for substantially the same reasons as discussed in Ground 2, 18[b].

d. 27[c]

“a scanner for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;”

319. In my opinion, AAPA discloses 27[c] for substantially the same reasons as discussed in Ground 2, 18[c].

e. 27[d]

“a run-length coder for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

320. In my opinion, AAPA discloses 27[d] for substantially the same reasons as discussed in Ground 2, 18[d].

f. 27[e]

“a context-based coder for assigning the second numbers to one of a plurality of contexts representative of the second numbers, and operative to assign the second number of a number pair to a context at least partly in dependence on the first number of the number pair.”

321. In my opinion, AAPA in view of VCEG-L13 discloses 27[e] for substantially the same reasons as discussed for Ground 3, 10[b].

10. Claim 28

“A system according to claim 27, wherein said context-based coder is a context-based arithmetic coder.”

322. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 11.

11. Claim 29

“A system according to claim 27, wherein said context-based coder is a context-based binary arithmetic coder.”

323. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 12.

12. Claim 30

“A system according to claim 29, wherein the context-based coder is operative to map the second numbers of said number pairs to a set of bins, each of which can be assigned to one of either a first or a second value.”

324. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 13.

13. Claim 31

“A system according to claim 30, wherein the context-based coder is operative to map a number to a given one of said set of bins by assigning the bin to the first value.”

325. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 14.

14. Claim 32

“A system according to claim 30, wherein the first value is 1 and the second value is 0.”

326. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 15.

15. Claim 33

“A system according to claim 30, wherein the context-based coder is operative to assign the second number of a number pair to a context at least partly in dependence on the bin to which the second number of the number pair is mapped.”

327. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 16.

16. Claim 34

“A system according to claim 27, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.”

328. In my opinion, AAPA discloses the additional limitations of this claim for substantially the same reasons as discussed in Ground 3, Claim 17.

17. Claim 36

g. 36[pre]

“A computer program embodied in a computer readable medium comprising:”

329. In my opinion, AAPA discloses 36[pre] for substantially the same reasons as discussed in Ground 3, 27[pre].

h. 36[a]

“machine-readable program code for dividing an image into a plurality of blocks having a plurality of pixels, each pixel having a pixel value;”

330. In my opinion, AAPA discloses 36[a] for substantially the same reasons as discussed in Ground 3, 27[a].

i. 36[b]

“machine-readable program code for performing a transform coding operation on a block of pixel values to produce a corresponding block of transform coefficient values;”

331. In my opinion, AAPA discloses 36[b] for substantially the same reasons as discussed in Ground 3, 27[b].

j. 36[c]

“machine-readable program code for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;”

332. In my opinion, AAPA discloses 36[c] for substantially the same reasons as discussed in Ground 3, 27[c].

k. 36[d]

“machine-readable program code for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and”

333. In my opinion, AAPA discloses 36[d] for substantially the same reasons as discussed in Ground 3, 27[d].

a. 36[e]

“machine-readable code for assigning the second numbers to one of a plurality of contexts representative of the second numbers such that the second number of a number pair is assigned to a context at least partly in dependence on the first number of the number pair.”

334. In my opinion, AAPA in view of Savatier discloses 36[e] for substantially the same reasons discussed for Ground 3, 27[e].

18. Claim 38

a. 38[pre]

“A method of context-based arithmetic encoding in which an array of data symbols is represented with a code-word, the method comprising:”

335. In my opinion, AAPA discloses 38[pre] for substantially the same

reasons as discussed in Ground 2, 37[pre]. As discussed for Ground 2, 37[pre], AAPA discloses this limitation because it describes a method of context-based arithmetic encoding in which a two-dimensional arrangement of quantized DCT coefficient values (*array of data symbols*) is represented with a one-dimensional arrangement of values (e.g., indicated in Figure 6b), such as “70” (*a code-word*). EX1001, 14:46-53, FIG. 6b.

b. 38[a]

“representing the data symbols in said array as number pairs comprising a first number and a second number;”

336. In my opinion, AAPA discloses 38[a] for substantially the same reasons discussed for Ground 3, 10[a].

c. 38[b]⁵

“assigning the second number of a number pair to a context selected from a plurality of contexts representative of the second numbers at least partly in dependence on the first number of the number pair.”

337. In my opinion, AAPA in view of VCEG-13 teaches 38[b] for substantially the same reasons discussed for Ground 3, 10[b].

19. Claim 40

a. 40[pre]

“A context-based arithmetic encoder operative to represent an

⁵ This claim language reflects a correction made in the June 21, 2005 Certificate of Correction. EX1002, 113.

array of data symbols with a code-word and comprising:”

338. In my opinion, AAPA discloses 40[pre] for substantially the same reasons as Ground 2, 39[pre].

b. 40[a]

“a run-level coder for representing the data symbols in said array as number pairs comprising a first number and a second number;”

339. In my opinion, AAPA discloses 40[a] for substantially the same reasons as discussed in Ground 2, 18[d].

c. 40[b]

“a context assigner for assigning the second number of a number pair to a context selected from a plurality of contexts representative of the second numbers at least partly in dependence on the first number of the number pair.”

340. In my opinion, AAPA in view of VCEG-L13 discloses 40[b] for substantially the same reasons as discussed in Ground 3, 10[b].

IX. CONCLUSION

341. I currently hold the opinions expressed in this Declaration. But my analysis may continue, and I may acquire additional information and/or attain supplemental insights that may result in added observations.

APPENDIX A – MATERIALS CONSIDERED

No.	Exhibit
1001	U.S. Patent No. 6,856,701 (“’701 patent”)
1002	File History of the ’701 patent
1005	U.S. Patent No. 5,818,877 (“Tsai”)
1006	International Telecommunication Union-Telecommunication Sector (“ITU-T”), <i>VLC concatenation for Intra coding</i> (2001), https://www.itu.int/wftp3/av-arch/video-site/0101_Eib/VCEG-L28.doc (“VCEG-L28”)
1007	ITU-T, <i>Adaptive Codes for H.26L</i> (2001), https://www.itu.int/wftp3/av-arch/video-site/0101_Eib/VCEG-L13.doc (“VCEG-L13”)
1008	U.S. Patent No. 5,400,075 (“Savatier”)
1009	ITU-T, Video coding for low bit rate communication (ITU-T H.263 Series H: Audiovisual and Multimedia Systems, Infrastructure of audiovisual services – Coding of moving video) (1996) (“H.263”)
1010	ITU-T, H.26L Test Model Long Term Number 4 (TML-4) draft0 (2000), https://www.itu.int/wftp3/av-arch/video-site/0005_Osa/q15j72d0.doc (“TML4”)
1011	ITU-T, <i>Use of adaptive switching between two VLCs for intra luma coefficients</i> (2000), https://www.itu.int/wftp3/av-arch/video-site/0008_Por/q15k30.doc (“Q15-K-30”)
1013	ITU-T, <i>Entropy Coding Efficiency of H.26L</i> (2000), https://www.itu.int/wftp3/av-arch/video-site/0005_Osa/q15j57.doc (“Q15-J-57”)
1014	Gary J. Sullivan, Thomas Wiegand, and Thomas Stockhammer, <i>Using the Draft H.26L Video Coding Standard for Mobile Applications</i> , IEEE, pp. 573-576 (2001) (“Sullivan”)

No.	Exhibit
1015	Ushashree, <i>Video Compression Based on RMB</i> , International Journal of Electronics Communication and Computer Engineering (2013) (“Ushashree”)
1016	Iain Richardson, <i>Video Codec Design: Developing Image and Video Compression Systems</i> (2002) (“Richardson”)
1017	Markus Flierl, Thomas Wiegand, and Bernd Girod, <i>Multihypothesis Pictures for H.26L</i> , Proceedings of the IEEE International Conference on Image Processing, Vol. 3, pp. 526-529 (2001) (“Flierl”)
1018	“Transfer of ftp://standards.pictel.com to ITU,” https://www.itu.int/wftp3/av-arch/Transfer%20of%20standards.pictel.com%20to%20ITU.htm (2003)
1019	Didier Le Gall, <i>MPEG: A Video Compression Standard For Multimedia Applications</i> , Communications of the ACM, Vol. 34, No. 4, pp. 46-58 (1991) (“LeGall”)
1020	Witten, Ian H., Radford M. Neal, and John G. Cleary, <i>Arithmetic Coding For Data Compression</i> , Communications of the ACM, Vol. 30, No. 6, pp. 520-540 (1987) (“Witten”)
1022	Martyn J. Riley and Iain E.G. Richardson, <i>Digital Video Communications</i> , Artech House, Inc. (1997) (“Riley”)

APPENDIX B – LISTING OF CHALLENGED CLAIMS

Reference	Limitation
Claim 1	
1[pre]	A method of image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values being scanned in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order, the method comprising the steps of:
1[a]	representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number; and
1[b]	assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of a second number pair.
Claim 2	
2	A method according to claim 1, wherein the first number of the first number pair is assigned to a context at least partly in dependence on the first number of a preceding number pair in the scanning order.
Claim 3	
3	A method according to claim 1, wherein said plurality of contexts are contexts of a context-based arithmetic coder.
Claim 4	
4	A method according to claim 1, wherein said plurality of contexts are contexts of a context-based binary arithmetic coder.

Reference	Limitation
Claim 5	
5	A method according to claim 4, further comprising the step of: mapping the first numbers of said number pairs to a set of bins, each of which can be assigned one of either a first or a second value.
Claim 6	
6	A method according to claim 5, wherein mapping of a number to a given one of said set of bins is indicated by assigning the bin to the first value.
Claim 7	
7	A method according to claim 5, wherein the first value is 1 and the second value is 0.
Claim 8	
8	A method according to claim 5, wherein the first number of a number pairs is assigned to a context at least partly in dependence on the bin to which the first number of said number pair is mapped.
Claim 9	
9	A method according to claim 1, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.
Claim 10	
10[pre]	A method of image coding, wherein an image is divided in an encoder into a plurality of blocks having a plurality of pixels, each pixel having a pixel value, and a transform coding operation is performed on a block of pixels to produce a corresponding block of transform coefficient values, the block of transform coefficient values is scanned in a given scanning order to produce a scanned

Reference	Limitation
	array of coefficient values arranged according to the scanning order, the method comprising the steps of:
10[a]	representing the coefficient values in the scanned array by a plurality of number pairs, each of said number pairs having a first number and a second number; and
10[b]	assigning the second numbers to one of a plurality of contexts representative of the second numbers such that the second number of a number pair is assigned to a context at least partly in dependence on the first number of the number pair.
Claim 11	
11	A method according to claim 10, wherein said plurality of contexts are contexts of a context-based arithmetic coder.
Claim 12	
12	A method according to claim 10, wherein said plurality of contexts are contexts of a context-based binary arithmetic coder.
Claim 13	
13	A method according to claim 12, further comprising the step of: mapping the second numbers of said number pairs to a set of bins, each of which can be assigned one of either a first value or a second value.
Claim 14	
14	A method according to claim 13, wherein mapping of a number to a given one of said set of bins is indicated by assigning the bin to the first value.
Claim 15	
15	A method according to claim 13, wherein the first value is 1 and the second value is 0.

Reference	Limitation
Claim 16	
16	A method according to claim 13, wherein the second number of a number pair is assigned to a context at least partly in dependence on the bin to which the second number of the number pair is mapped.
Claim 17	
17	A method according to claim 10, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.
Claim 18	
18[pre]	A system for image coding comprising:
18[a]	an input for receiving an image as a plurality of blocks having a plurality of pixels, each pixel having a pixel value;
18[b]	a transform coder for performing a transform coding operation on a block of pixels to produce a corresponding block of transform coefficient values;
18[c]	a scanner for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values, arranged according to the scanning order;
18[d]	a run-level coder for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and
18[e]	a context-based coder for assigning the first numbers to one of a plurality of contexts representative of the first numbers and operative to assign the first number of a first number pair to a context at least partly in dependence on a first number of a second number pair.

Reference	Limitation
Claim 19	
19	A system according to claim 18, wherein said context-based coder is a context-based arithmetic coder.
Claim 20	
20	A system according to claim 18, wherein said context-based coder is a context-based binary arithmetic coder.
Claim 21	
21	A system according to claim 20, wherein the context-based coder is operative to map the first numbers of said number pairs to a set of bins, each of which can be assigned to one of either a first or a second value.
Claim 22	
22	A system according to claim 21, wherein the context-based coder is operative to map a number to a given one of said set of bins by assigning the bin to the first value.
Claim 23	
23	A system according to claim 21, wherein the first value is 1 and the second value is 0.
Claim 24	
24	A system according to claim 21, wherein the context-based coder is operative to assign the first number of a number pair to a context at least partly in dependence on the bin to which the first number of said number pair is mapped.
Claim 25	
25	A system according to claim 18, wherein the context-based coder is operative to assign the first number of the first number pair to a

Reference	Limitation
	context at least partly in dependence on the first number of a preceding number pair in the scanning order.
Claim 26	
26	A system according to claim 18, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.
Claim 27	
27[pre]	A system for image coding comprising:
27[a]	an input for receiving an image as a plurality of blocks having a plurality of pixels, each pixel having a pixel value;
27[b]	a transform coder for performing a transform coding operation on a block of pixels to produce a corresponding block of transform coefficient values;
27[c]	a scanner for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;
27[d]	a run-length coder for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and
27[e]	a context-based coder for assigning the second numbers to one of a plurality of contexts representative of the second numbers, and operative to assign the second number of a number pair to a context at least partly in dependence on the first number of the number pair.
Claim 28	
28	A system according to claim 27, wherein said context-based coder is a context-based arithmetic coder.

Reference	Limitation
Claim 29	
29	A system according to claim 27, wherein said context-based coder is a context-based binary arithmetic coder.
Claim 30	
30	A system according to claim 29, wherein the context-based coder is operative to map the second numbers of said number pairs to a set of bins, each of which can be assigned to one of either a first or a second value.
Claim 31	
31	A system according to claim 30, wherein the context-based coder is operative to map a number to a given one of said set of bins by assigning the bin to the first value.
Claim 32	
32	A system according to claim 30, wherein the first value is 1 and the second value is 0.
Claim 33	
33	A system according to claim 30, wherein the context-based coder is operative to assign the second number of a number pair to a context at least partly in dependence on the bin to which the second number of the number pair is mapped.
Claim 34	
34	A system according to claim 27, wherein said first number is indicative of a non-zero coefficient value and said second number is indicative of a number of consecutive zero coefficient values preceding said non-zero coefficient value.

Reference	Limitation
Claim 35	
35[pre]	A computer program embodied in a computer readable medium comprising:
35[a]	machine-readable program code for dividing an image into a plurality of blocks having a plurality of pixels, each pixel having a pixel value;
35[b]	machine-readable program code for performing a transform coding operation on a block of pixel values to produce a corresponding block of transform coefficient values
35[c]	machine-readable program code for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;
35[d]	machine-readable program code for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and
35[e]	machine-readable program code for assigning the first numbers to one of a plurality of contexts representative of the first numbers such that the first number of a first number pair is assigned to a context at least partly in dependence on a first number of second number pair.
Claim 36	
36[pre]	A computer program embodied in a computer readable medium comprising:
36[a]	machine-readable program code for dividing an image into a plurality of blocks having a plurality of pixels, each pixel having a pixel value;
36[b]	machine-readable program code for performing a transform coding operation on a block of pixel values to produce a corresponding block of transform coefficient values;

Reference	Limitation
36[c]	machine-readable program code for scanning the block of transform coefficient values in a given scanning order to produce a scanned array of coefficient values arranged according to the scanning order;
36[d]	machine-readable program code for representing the coefficient values in the scanned array by a plurality of number pairs, said number pairs having a first number and a second number; and
36[e]	machine-readable code for assigning the second numbers to one of a plurality of contexts representative of the second numbers such that the second number of a number pair is assigned to a context at least partly in dependence on the first number of the number pair.
Claim 37	
37[pre]	A method of context-based arithmetic encoding in which an array of data symbols is represented with a code-word, the method comprising:
37[a]	representing the data symbols in said array as number pairs comprising a first number and a second number; and
37[b]	assigning the first number of a first number pair to a context selected from a plurality of contexts representative of the first numbers at least partly in dependence on a first number of a second number pair.
Claim 38	
38[pre]	A method of context-based arithmetic encoding in which an array of data symbols is represented with a code-word, the method comprising:
38[a]	representing the data symbols in said array as number pairs comprising a first number and a second number;

Reference	Limitation
38[b] ⁶	assigning the second number of a number pair to a context selected from a plurality of contexts representative of the second numbers at least partly in dependence on the first number of the number pair.
Claim 39	
39[pre]	A context-based arithmetic encoder operative to represent an array of data symbols with a code-word and comprising:
39[a]	a run-level coder for representing the data symbols in said array as number pairs comprising a first number and a second number;
39[b]	a context assigner for assigning the first number of a first number pair to a context selected from a plurality of contexts representative of the first numbers at least partly in dependence on a first number of a second number pair.
Claim 40	
40[pre]	A context-based arithmetic encoder operative to represent an array of data symbols with a code-word and comprising:
40[a]	a run-level coder for representing the data symbols in said array as number pairs comprising a first number and a second number;
40[b]	a context assigner for assigning the second number of a number pair to a context selected from a plurality of contexts representative of the second numbers at least partly in dependence on the first number of the number pair.

⁶ This claim language reflects a correction made in the June 21, 2005 Certificate of Correction. EX1002, 113.