

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

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 6,856,701**

Mail Stop PATENT BOARD
Patent Trial and Appeal Board
U.S. Patent & Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

[illegible]

13.	Claim 11	41
14.	Claim 12	41
15.	Claim 13	41
16.	Claim 14	41
17.	Claim 17	42
18.	Claim 18	42
19.	Claim 19	48
20.	Claim 20	48
21.	Claim 21	48
22.	Claim 22	48
23.	Claim 23	48
24.	Claim 24	48
25.	Claim 25	48
26.	Claim 26	49
27.	Claim 27	49
28.	Claim 28	50
29.	Claim 29	50
30.	Claim 30	50
31.	Claim 31	50
32.	Claim 34	50
33.	Claim 35	51
34.	Claim 36	52
35.	Claim 37	53
36.	Claim 38	54
37.	Claim 39	54
38.	Claim 40	55
B.	Ground 1B: Obviousness over Tsai, VCEG-L28, VCEG-L13 (Claims 15-16, 32-33)	55
1.	VCEG-L13 Overview	55

2.	Claim 15	56
3.	Claim 16	59
4.	Claim 32	59
5.	Claim 33	59
C.	Ground 2: Obviousness over AAPA and Savatier (Claims 1-9, 18-26, 35, 37, 39)	59
1.	AAPA Overview	59
2.	Savatier Overview	60
3.	Claim 1	63
4.	Claim 2	76
5.	Claim 3	76
6.	Claim 4	77
7.	Claim 5	77
8.	Claim 6	78
9.	Claim 7	79
10.	Claim 8	80
11.	Claim 9	81
12.	Claim 18	81
13.	Claim 19	86
14.	Claim 20	87
15.	Claim 21	87
16.	Claim 22	87
17.	Claim 23	87
18.	Claim 24	87
19.	Claim 25	87
20.	Claim 26	87
21.	Claim 35	88
22.	Claim 37	89
23.	Claim 39	90

D.	Ground 3: Obviousness over AAPA and VCEG-L13 (Claims 10-17, 27-34, 36, 38, 40)	91
1.	Claim 10	91
2.	Claim 11	94
3.	Claim 12	94
4.	Claim 13	94
5.	Claim 14	94
6.	Claim 15	95
7.	Claim 16	95
8.	Claim 17	95
9.	Claim 27	96
10.	Claim 28	97
11.	Claim 29	97
12.	Claim 30	97
13.	Claim 31	97
14.	Claim 32	97
15.	Claim 33	97
16.	Claim 34	97
17.	Claim 36	98
18.	Claim 38	99
19.	Claim 40	99
VII.	DISCRETIONARY DENIAL WOULD BE INAPPROPRIATE	100
VIII.	CONCLUSION	100
IX.	MANDATORY NOTICES	101
A.	Real Party in Interest	101
B.	Related Matters	101
C.	Notice of Counsel and Service Information	102

EXHIBIT LIST

No.	Exhibit
1001	U.S. Patent No. 6,856,701 (“’701 patent”)
1002	File History of the ’701 patent
1003	Declaration of Dr. Eli Saber
1004	Curriculum Vitae of Dr. Eli Saber
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”)
1012	Original Complaint, <i>Nokia Techs. Oy v. Amazon.com, Inc.</i> , No. 1:23-cv-01236 (D. Del. Oct. 31, 2023), ECF No. 1
1013	ITU-T, <i>Entropy Coding Efficiency of H.26L</i> (2000),

No.	Exhibit
	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")
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")
1021	Declaration of Dr. Stephan Wenger ("Wenger")
1022	Martyn J. Riley and Iain E.G. Richardson, <i>Digital Video Communications</i> , Artech House, Inc. (1997) ("Riley")
1023	United States District Courts — National Judicial Caseload Profile, UNITED STATES COURTS (June 30, 2024), <i>available at</i>

No.	Exhibit
	https://www.uscourts.gov/sites/default/files/data_tables/fcms_na_distprofile0630.2024.pdf

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 array of coefficient values arranged according to the scanning order,

Reference	Limitation
	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 context at least partly in dependence on the first number of a

Reference	Limitation
	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] ¹	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.

¹ This claim language reflects a correction made in the June 21, 2005 Certificate of Correction. EX1002, 113.

Amazon.com, Inc. and Amazon.com Services LLC (“Amazon” or “Petitioner”) petition for *inter partes* review (“IPR”) of claims 1-40 (“Challenged Claims”) of U.S. Patent No. 6,856,701 (“’701 patent”; EX1001).

I. GROUNDS FOR STANDING

Petitioner certifies the ’701 patent is available for IPR, and Petitioner is not barred or estopped from requesting IPR on the grounds herein. Petitioner files this Petition within one year of service of Patent Owner Nokia Technologies Oy’s (hereafter, “PO”) district court complaint. EX1012.

II. IDENTIFICATION OF CHALLENGE

A. Prior Art

Petitioner respectfully requests cancellation of the Challenged Claims over:

Prior Art
U.S. Patent No. 5,818,877, issued on October 6, 1998; prior art under at least 35 U.S.C. §102(b) ² (“Tsai”; EX1005)
VLC Concatenation for Intra Coding, published on January 3, 2001; prior art under at least §102(a) (“VCEG-L28”; EX1006)

² September 14, 2001 is the earliest claimed priority date of the ’701 patent. The references herein qualify as prior art based on that date. Petitioner does not concede, however, that any Challenged Claim is entitled to the asserted priority date.

Adaptive Codes for H.26L, published on January 8, 2001; prior art under at least §102(a) (“VCEG-L13”; EX1007)

U.S. Patent No. 5,400,075, issued on March 21, 1995; prior art under at least 35 U.S.C. §102(b) (“Savatier”; EX1008)

B. Grounds for Challenge

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	Applicant Admitted Prior Art (“AAPA”), Savatier
3	§103	10-17, 27-34, 36, 38, 40	AAPA, VCEG-L13

III. ’701 PATENT OVERVIEW

A. Technology Background

Image compression/coding is essential for efficiently storing and transmitting image and video data. LeGall, 47-54; EX1003, ¶¶37-40. To reduce the data required to code an image, pixel values are scanned and converted to coefficient values using techniques like discrete cosine transform (“DCT”). EX1001, 2:34-56; Riley, 21; EX1003, ¶¶41-44. Coefficient values are then quantized, resulting in a sequential list of integers represented using pairs of values referred to as “level” and “run,” where level is the value of a non-zero coefficient and run is the number of consecutive zero coefficients

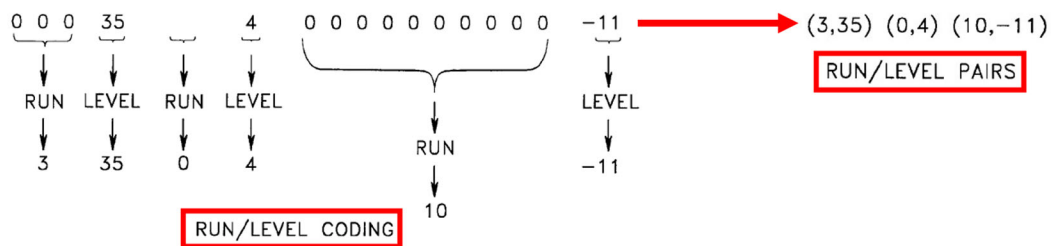
preceding the non-zero coefficient. EX1001, 6:6-17; EX1003, ¶¶45-52. An exemplary list of integers is provided in FIG. 4B of Tsai (below). Tsai, 9:49-56.

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

SEQUENTIAL INTEGER LIST

Tsai, FIG. 4B (simplified).

Run/level coding converts the list into pairs of run and level values, as shown in FIGS. 4C-4D (below). Tsai, 9:56-10:9.



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

Run and level values undergo further compression by entropy coding, which assigns shorter codewords to values that occur more frequently. EX1001, 3:3-18, 6:18-32; EX1003, ¶¶53-55. A type of entropy coding known as Context-Based Adaptive Binary Arithmetic Coding (“CABAC”) adapts to the statistical properties of the coded data. VCEG-L13, 1-2, 6-7; EX1001, 11:63-12:2; EX1003, ¶¶56-61. Specifically, run/level values are coded as sequences of binary decisions (“bins”) based on a probability/statistical model (“context”) associated with each bin. VCEG-L13, 1-2, 5; EX1001, 12:42-13:28; EX1003, ¶¶57-59. 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; EX1003, ¶60.

Before the '701 patent, the Video Coding Experts Group ("VCEG") under the International Telecommunication Union-Telecommunication Sector ("ITU-T") made significant progress in developing CABAC as part of H.26L, which became the H.264/Advanced Video Coding ("AVC") compression standard. EX1003, ¶61.

B. Disclosure of the '701 Patent

The '701 patent relates to image and video compression using CABAC, and particularly the coding of run and level values for blocks of quantized DCT transform coefficients. EX1001, 1:10-13, 15:47-61; EX1003, ¶62.

1. AAPA

The Background of the Invention section of the '701 patent describes the known art as of the '701 patent's earliest priority date. EX1001, 1:17-15:61; EX1003, ¶63.

"Admissions [i.e., AAPA] are 'permissible evidence in an [IPR] for establishing the background knowledge possessed by a person of ordinary skill in the art' and 'provide a factual foundation as to what a skilled artisan would have known at the time of invention.'" *Updated Guidance on the Treatment of*

Statements of the Applicant in the Challenged Patent in Inter Partes Reviews Under §311 (hereinafter, “*AAPA-Guidance*”), 3 (June 9, 2022) (citation omitted). “[AAPA] may include statements in the specification of the challenged patent such as ‘It is well known that...,’ ‘It is well understood that...,’ or ‘One of skill in the art would readily understand that...,’ or may describe technology as ‘prior art,’ ‘conventional,’ or ‘well-known.’” *AAPA-Guidance*, 4 (citation omitted).

a. Generic hybrid video encoder/decoder

The Background describes a prior art “generic hybrid video encoder” (shown in FIG. 1) and corresponding decoder (shown in FIG. 2) using terminology indicating it is AAPA. EX1001, 3:45-11:35; EX1003, ¶66. For example, the ’701 patent explains that “[t]he 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.” EX1001, 11:43-47. 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[.]” EX1001, 11:37-40; EX1003, ¶70.

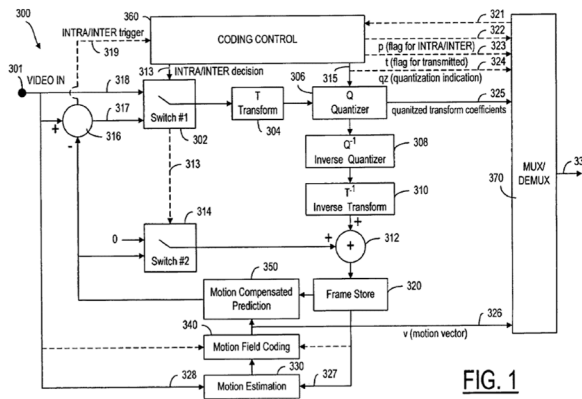


FIG. 1

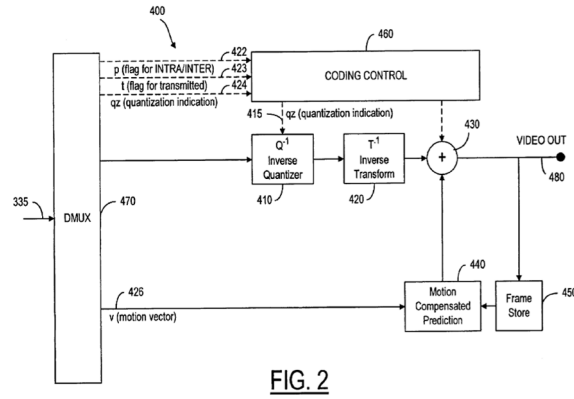


FIG. 2

EX1001, Figure 1 and Figure 2.

The generic hybrid video encoder/decoder of FIGS. 1-2 and associated disclosure are thus AAPA. *See AAPA-Guidance*, 4-6.

b. Block-based transform coding

The Background uses the AAPA generic hybrid video encoder/decoder of FIGS. 1 and 2 “to illustrate principles of block-based transform coding” (EX1001, 4:25-31) that were known in the art. *See* EX1001, 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.”), 4:17-19 (“Generally, INTRA-coded frames are produced by applying block-based transform coding directly to the pixel values of the frame to be coded.”); EX1003, ¶67. Block-based transform coding is thus also AAPA. *See AAPA-Guidance*, 4-6.

c. Run/level coding

The Background also describes run/level coding 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 Background explains that ITU-T H.26L uses run/level coding. EX1001, 12:7-26, 14:2-16:24, 16:62-64; EX1003, ¶¶68. Run/level coding is thus also AAPA. *See AAPA-Guidance*, 4-6.

d. Entropy coding

The Background 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 “different methods” for “implement[ing] entropy coding”); EX1003, ¶¶64, 69-70. Entropy coding is thus also AAPA. *See AAPA-Guidance*, 4-6.

e. CABAC

The Background explains that “[**CABAC**]...is a form of binary arithmetic coding which continually adapts to the statistical properties of the information

³ All emphases added unless otherwise stated.

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 Witten (EX1020)); EX1003, ¶¶65, 71-74. The ’701 patent further describes an exemplary CABAC encoder 700 and decoder 800 with reference to FIGS. 4 and 5 respectively (below), both of which are labeled “PRIOR ART.” EX1001, 12:35-14:4. The ’701 patent describes operation of CABAC with reference to FIGS. 6a-6d. EX1001, 14:46-15:53. According to the ’701 patent, CABAC was introduced into “ITU-T recommendation H.26L[,]. . . the latest in a family of video coding standards developed by the [ITU]. . . for video coding at very low bit rates[.]” EX1001, 11:37-40, 12:15-18; *see id.*, 14:5-15:61. CABAC is thus also AAPA. *See AAPA-Guidance*, 4-6.

2. Alleged Problem in the Prior Art and Purported Solution

The ’701 patent alleges CABAC is “still not optimal with regard to coding efficiency.” EX1001, 15:54-58. The patent purportedly addresses this problem using “relationships [that] exist between the run and level values associated with DCT transform coefficients. . . to construct improved context models which enable the CABAC method to operate with improved coding efficiency when applied to the run and level values.” EX1001, 16:2-10; EX1003, ¶75.

The Challenged Claims, however, recite nothing more than what was already known, including the allegedly novel step of determining a probability

characteristic (i.e., a 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 of the same pair. EX1003, ¶76.

IV. LEVEL OF ORDINARY SKILL

A person of ordinary skill in the art (“POSITA”) at the time 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 2-3 years of practical experience with video and/or image processing or coding. More education could substitute for experience, and vice versa. EX1003, ¶¶22-24.

V. CLAIM CONSTRUCTION

Petitioner is unaware of any prior claim construction determination for the ’701 patent. 37 C.F.R. §42.100(b). Here, given the close correlation between the prior-art references and the Challenged Claims, Petitioner believes that no construction is necessary to evaluate the grounds herein.

VI. DETAILED EXPLANATION OF GROUNDS

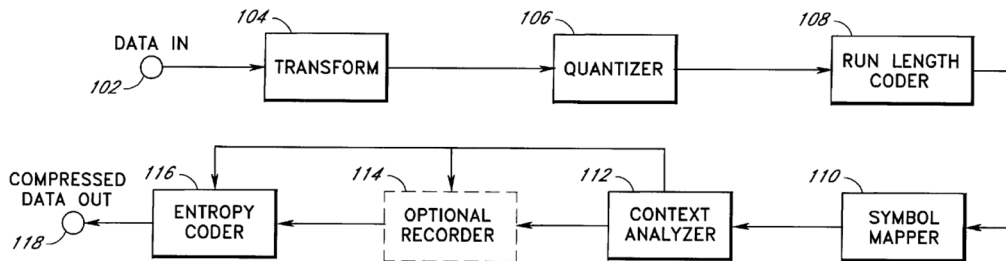
A. Ground 1A: Obviousness over Tsai and VCEG-L28 (Claims 1-14, 17-31, 34-40)

1. Tsai Overview

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; EX1003, ¶80.

FIG. 1 (below) shows a data compression system in an embodiment. Tsai, 5:33-34, 6:60-61; EX1003, ¶¶81-83. Digital data is input to transform processor 104 to generate transformed data, e.g., DCT data. Tsai, 7:1-9. The transformed data is subsequently passed to quantizer 106 and then to run-length coder 108 (Tsai, 7:10-44), which orders the quantized data into a one-dimensional list of integers and converts the list into a sequence of (run, level) pairs. Tsai, 7:45-64.



Tsai, FIG. 1.

Tsai explains that “the way in which this [(run, level)] information is mapped for subsequent entropy coding plays a critical role in the performance of the compression method,” and Tsai thus discloses techniques to compress the (run, level) pairs. Tsai, 2:44-55. 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. Tsai’s mapping “eliminates

explicit representation of the [most significant bit] MSB for both run values and level values...[thereby] reduc[ing] the number of output symbols, and contribut[ing] directly to greater efficiency.” Tsai, 4:39-43. “Furthermore, the presence of two contexts is used to increase the performance of the entropy coding step that follows the mapping to symbols.” Tsai, 4:44-50.

When mapping the (run, level) pairs, Tsai notes that “[t]o improve coding efficiency, it is desirable that shorter words⁴ be used to represent run values that have a higher probability of occurrence.” Tsai, 10:35-39. “It 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.

For example, Tsai’s FIG. 5 (below) shows Symbol Map 1 and Symbol Map 2 for mapping symbol groups to run values. Tsai, 10:39-41. In this mapping, small run values are associated with short words, desirable in systems “more likely to produce runs that are small” such as “typical image and sound data.” Tsai, 10:35-41; EX1003, ¶¶84-93.

⁴ Tsai defines “word” as “a sequence of symbols...derive[d] from a symbol set[.]” Tsai, 4:19-25, 10:25-33; EX1003, ¶53 n.2.

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		----	+++-
.	.	.	.
.	.	.	.
.	.	.	.

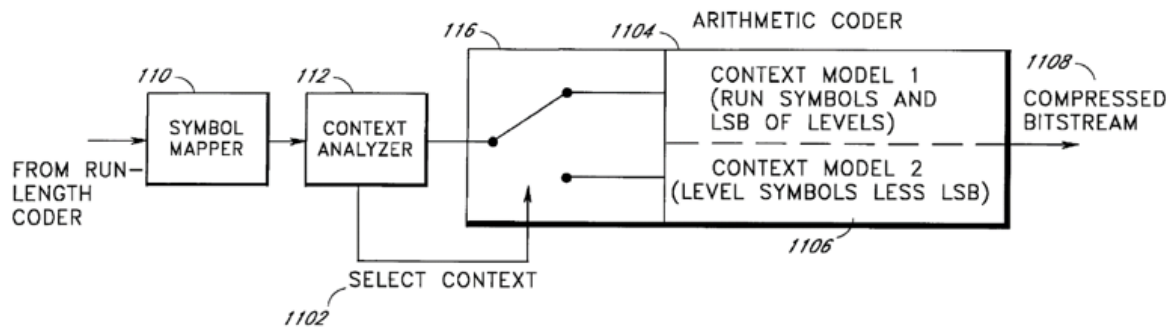
Tsai, FIG. 5.

Similarly, for level values, “additional efficiency in entropy coding can be gained when input symbols appear with unequal probability.” Tsai, 11:53-54. Tsai’s FIG. 7 (below) illustrates two symbol maps used to map symbol groups to level values. “Symbol Map 2 arranges the words so that the symbol ‘0’ occurs more frequently in words associated with small level values,” improving compression “in the case where run/level coded data contain a large percentage of low level values.” Tsai, 11:55-61.

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.

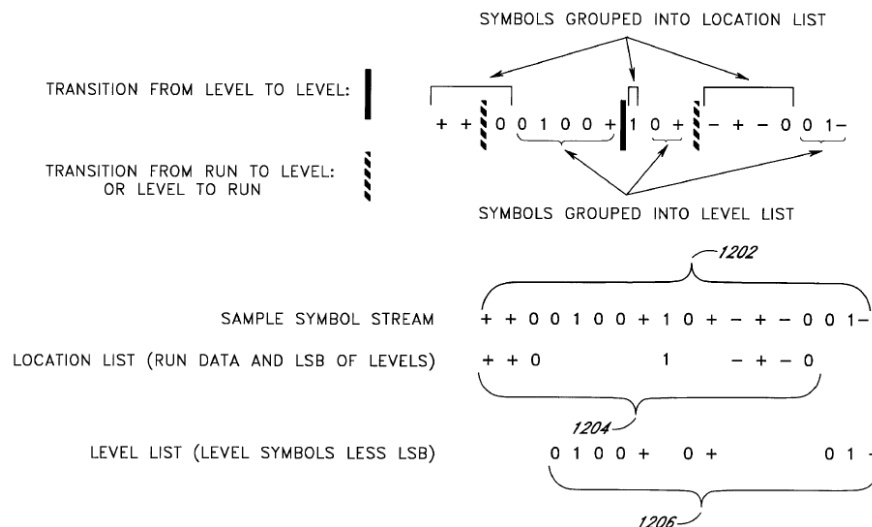
Tsai's FIG. 11 (below) shows the flow of data from symbol mapper 110 into context analyzer 112 and then into arithmetic coder 116. Tsai, 14:65-67. After symbol mapper 110 maps the (run, level) pairs into a symbol stream, "context analyzer 112 partitions the symbol stream into two different types of symbol 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." Tsai, 15:7-12.



Tsai, FIG. 11.

Tsai's FIG. 12 (below) shows an exemplary 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, 15:38-40.



Tsai, FIG. 12.

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 Overview

VCEG-L28 is a proposal from the VCEG under the ITU-T for modifying the H.26L standard. VCEG-L28, 1. On its face, VCEG-L28 indicates that it was generated on January 3, 2001 for the January 9-12, 2001 VCEG meeting in Eibsee, Germany. VCEG-L28, 1.

A reference is considered publicly accessible if “persons interested and ordinarily skilled in the subject matter or art, exercising reasonable diligence, can locate it.” *Acceleration Bay, LLC v. Activision Blizzard Inc.*, 908 F.3d 765, 772 (Fed. Cir. 2018) (citation omitted).

Dr. Stephan Wenger, a VCEG member who attended the Eibsee meeting, confirms that VCEG-L28 was submitted in connection with that meeting. Wenger, ¶¶12, 17. Dr. Wenger further explains that VCEG-L28 was made available on a public VCEG FTP server by January 3, 2001. Wenger, ¶¶18-38 (discussing EX1014-EX1018). Subscribers to the VCEG email reflector, including participants, students, and the public, were commonly notified by email when files were uploaded to the FTP folder. Wenger, ¶¶26-27. Files were frequently downloaded from the publicly accessible folder. Wenger, ¶¶20-22, 27-28. Persons interested in tracking the developments of the latest video coding standard regularly visited the VCEG sites to ensure that products and services they were developing were consistent with the standards under development. Wenger, ¶¶37-38. A POSITA would have known that proposals like VCEG-L28 included technologies potentially destined for the developing standard. Wenger, ¶37. Dr. Wenger personally shared VCEG work with colleagues and students during the relevant time. Wenger, ¶35. VCEG-L28 is thus §102(a) prior art to the '701 patent.

As discussed *supra* §III.A, entropy coding assigns shorter codewords to frequently occurring run/level values. *E.g.*, EX1001, 6:19-32. VCEG-L28 teaches that “coding improvements are possible by switching between two VLC tables” during the run/level coding process. VCEG-L28, §1; Q15-K-30, 1-2; EX1003,

¶¶94-97. 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 the two tables 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.

By using UVLC2 when coding a symbol, up to two bits can be saved to achieve better coding performance. VCEG-L28, §2. The decision to switch between UVLC and UVLC2 is 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.

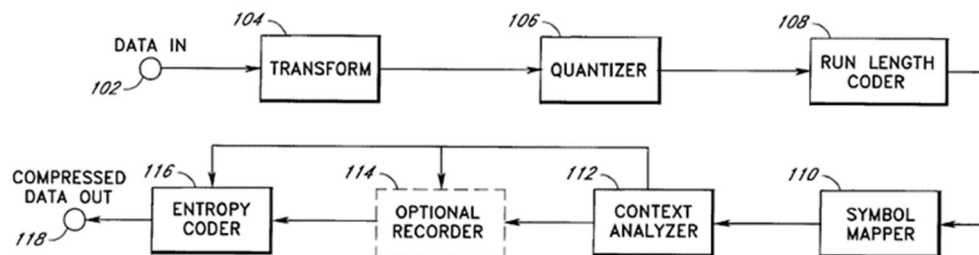
VCEG-L28 explains that use of UVLC2 provides coding improvements over the H.26L standard. VCEG-L28, §5; Q15-J-57, 3; EX1003, ¶98.

3. Claim 1

a. 1[pre]⁵

To the extent that it is limiting, Tsai teaches 1[pre].

Tsai teaches the JPEG and MPEG compression standards, each of which a POSITA would have understood to be *a method of image coding*. Tsai, 1:62-2:22; EX1003, ¶100. Tsai's FIG. 1 (below) shows “a data compression system which uses [Tsai's] invention[]...[that] can compress many types of data...for example, image, video, or sound.” Tsai, 6:60-67. A POSITA would have understood Tsai's data compression system to be *an encoder* because it compresses image and video data. EX1003, ¶100.



Tsai, FIG. 1.

⁵ A list of the Challenged Claims precedes the Introduction.

Transform processor 104 transforms “**8x8 pixel blocks** comprising an image.” Tsai, 7:1-9; *see id.*, 1:63-2:4 (JPEG), 2:11-22 (MPEG), 4:6-23. Specifically, DCT is “performed on 8x8 pixel blocks comprising an image[.]” Tsai, 2:1-2. Tsai thus teaches that *an image is divided in an encoder into a plurality of blocks having a plurality of pixels*. EX1003, ¶101. Tsai further recognizes that each pixel is represented using *a pixel value*. Tsai, 2:34-37; EX1003, ¶102.

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; EX1003, ¶¶42-44.

As discussed above, DCT is “**performed on 8x8 pixel blocks** comprising an image[.]” Tsai, 2:1-2. 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.

402

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

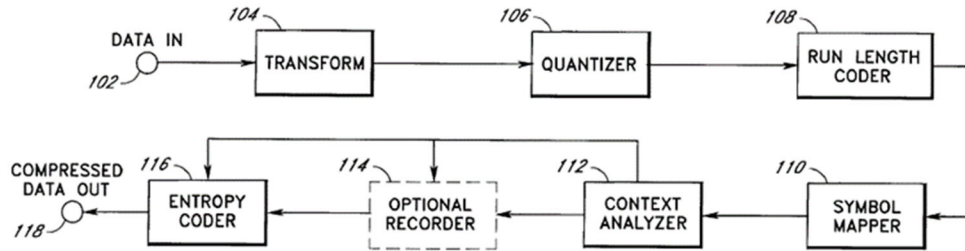
INPUT DATA ARRAY

Tsai, FIG. 4A.

A POSITA would have understood that Tsai's DCT operation transforms each 8x8 block of pixel values into a corresponding 8x8 block of DCT coefficients. See EX1003, ¶¶42-44. Tsai thus teaches "*a transform coding operation [i.e., DCT] is performed on a block of pixels to produce a corresponding block of transform coefficient values.*" EX1003, ¶¶103-104.

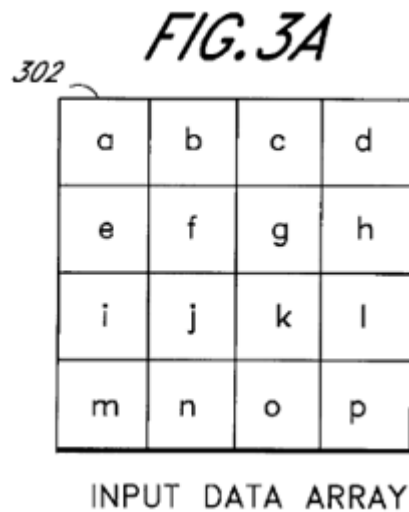
Tsai explains that transform processor 104 outputs transformed data corresponding to an 8x8 pixel block (*the block of transform coefficient values*) to quantizer 106, which "maps the transformed data into a smaller set of output values." Tsai, 7:10-13. "[T]he quantized data consist of positive and negative integers, and in some cases, a large percentage of these integers will be zero." Tsai, 7:41-43.

As shown in 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 (“First,...run-length coder 108 orders the quantized integer data into a one-dimensional list of integers.”); EX1003, ¶¶105-111.



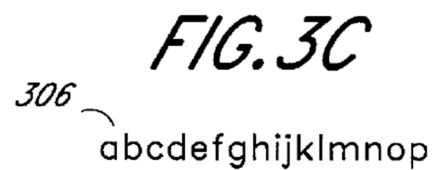
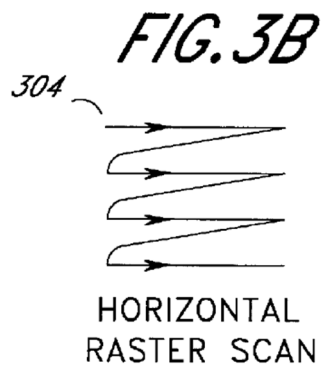
Tsai, FIG. 1.

“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.

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.

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.

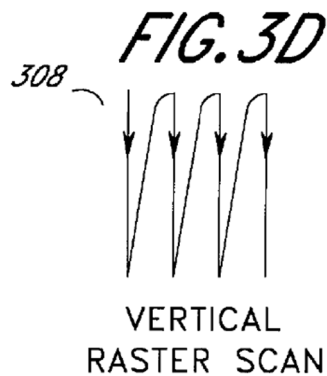


FIG. 3E

310

aeimbfjncgkodhlp

Tsai, FIGS. 3D-3E.

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.

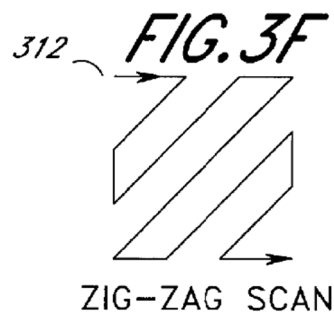


FIG. 3G

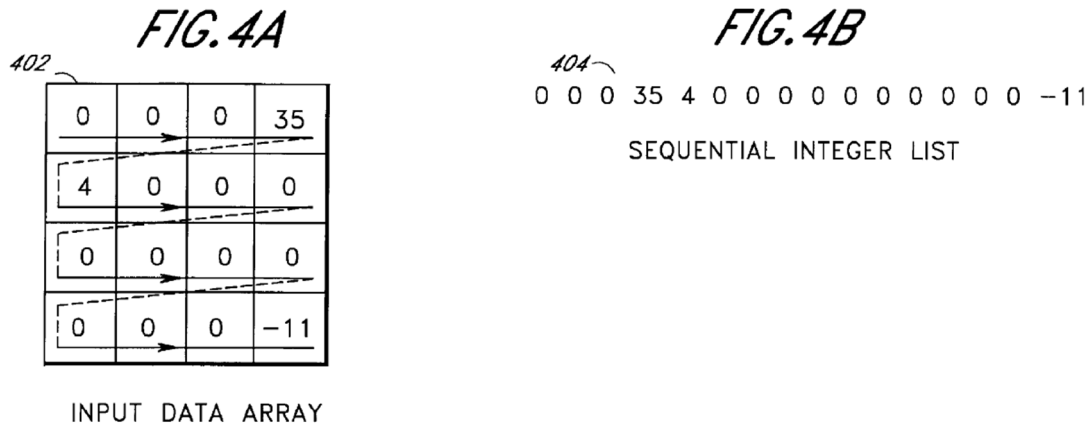
314

abeifcdgjmnkhlop

Tsai, FIGS. 3F-3G.

Furthermore, FIGS. 4A-4B 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. And 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. EX1003, ¶111.



Tsai, FIGS. 4A-4B.

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*. EX1003, ¶112.

b. 1[a]

Tsai teaches this limitation. After ordering the quantized integer data into a one-dimensional list of integers as discussed for 1[pre], “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.

Tsai’s FIG. 4D (below) shows the coefficient values arranged according to the scanned order represented as run/level pairs (*a plurality of number pairs*), each

having a level value (*a first number*) and a run value (*a second number*). Tsai, 10:5-9; EX1003, ¶¶113-117 (discussing H.263, §5.4.2, Table 13; TML4, Table 1, §4.3.4).

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

Tsai, FIG. 4D.

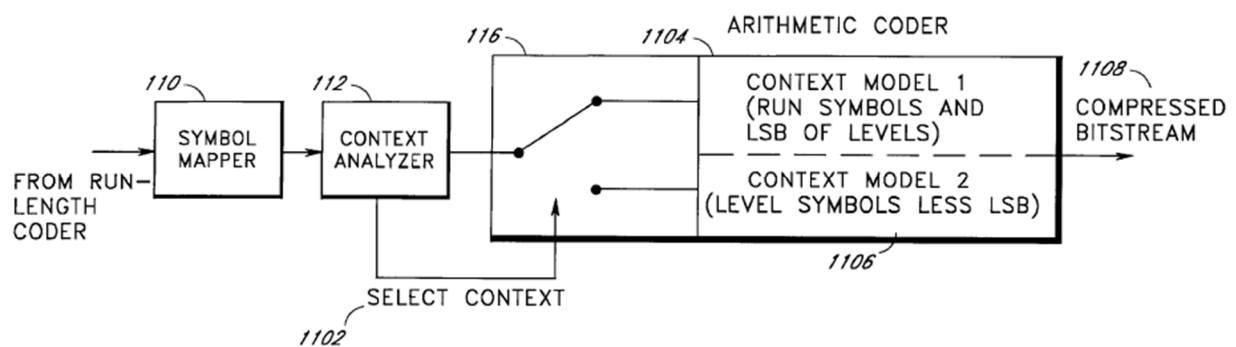
c. 1[b]

Tsai in view of VCEG-L28 teaches this limitation.

As discussed for 1[a], Tsai teaches that the coefficient values are represented as run/level pairs, each having a level value (*a first number*) and a run value (*a second number*).⁶ Tsai teaches the use of two “contexts” to optimize entropy coding by developing separate probability tables for each context, thereby adapting entropy codes based on changing symbol statistics. Tsai, 4:44-50. Tsai therefore teaches that different statistical characteristics correspond to different probability tables. EX1003, ¶118.

⁶ The '701 patent identifies the level value as the *first number* and the run value as the *second number*. See EX1001, 6:12-14.

Tsai's FIG. 11 (below) shows “[t]he symbol stream produced by the symbol mapper 110 is input to the context analyzer 112...[which] examines the input symbols and detects changes in context (run context or level context) indicated by the differences between the respective alphabetic symbols comprising run words and level words.” Tsai, 14:67-15:6.



Tsai, FIG. 11.

After partitioning the input symbol stream into two different types of symbol groups, when the context analyzer determines “a group of symbols comprising a level word with no LSB,” the context analyzer “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.

Tsai therefore teaches *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*). EX1003, ¶¶119-122.

During the symbol mapping stage, Tsai notes 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.

As shown in Tsai’s FIG. 7 (below), two symbol maps corresponding to different probability statistics can be used by the symbol mapper. EX1003, ¶¶123-124. 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, the symbol representations for the level value are different when different symbol maps are used. EX1003, ¶124. 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 (e.g., value 10), more ‘0’ bins are included in the output stream, which can be better compressed as compared to ‘1’ bins. EX1003, ¶124.

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).

In the same field of coding run/level pairs of transform coefficients, VCEG-L28 teaches that switching between two tables (UVLC and UVLC2) can lead to improved video coding efficiency. VCEG-L28, Title, §5. Specifically, as shown below, VCEG-L28 teaches using *previous_level*, which is the level value in the preceding value pair, to determine when to switch between UVLC and UVLC2 for symbol mapping. 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).

VCEG-L28 thus teaches using *previous_level*—i.e., the absolute value of the coefficient from a prior block or scan—as an indicator to switch between different tables, allowing the coder to better adapt to the nature of the data and leading to better compression. EX1003, ¶¶125-126.

Combination of Tsai and VCEG-L28: A POSITA implementing Tsai's symbol mapper would have been motivated to incorporate VCEG-L28's teaching of switching based on the previous level. In the combination, Tsai's context assignment (i.e., selection between Symbol Map 1 or Symbol Map 2) is based on the *previous_level* criteria taught in VCEG-L28. That is, selection of the symbol table based on the previous level affects the output symbol stream and thereby also affects the context assignment. EX1003, ¶127.

A POSITA would have been motivated to make the combination for several reasons.

First, a POSITA would have recognized that Tsai and VCEG-L28 both relate to improvements in the run/level coding process in video compression systems. Tsai, Abstract, 1:28-33, 2:53-55; VCEG-L28, Title, §5; *KSR, Int'l Co. v. Teleflex, Inc.*, 550 U.S. 398, 418 (2007). A POSITA would have been motivated to combine Tsai's and VCEG-L28's complementary coding techniques to further their common goal of improved coding efficiency. EX1003, ¶127.

Second, the combination would have resulted in an improved system because VCEG-L28's method of selecting between Symbol Map 1 and Symbol Map 2 based on the *previous_level* uses fewer bits to represent symbols that appear more frequently, thereby improving Tsai's coding efficiency. VCEG-L28, §2; EX1003, ¶¶128-129.

Third, 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 coder using symbol maps for context assignment), and the combination would yield predictable results (a coder having improved coding efficiency). EX1003, ¶130

Further, a POSITA would have reasonably expected success implementing VCEG-L28's teachings in Tsai's system because Tsai's system is configured to accommodate a two-table framework (e.g., Symbol Map 1 and Symbol Map 2). Tsai, 10:35-41, FIG. 5. A POSITA would have thus found Tsai's system to be

compatible with VCEG-L28's use of UVLC and UVLC2. VCEG-L28, §§1-2, 4, Table 1; EX1003, ¶130.

Thus, 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 at least partly depends upon the level value in a previous pair (*in dependence on a first number of a second number pair*). EX1003, ¶131.

4. Claim 2

Tsai in view of VCEG-L28 teaches the additional limitations of this claim. As discussed in 1[b], VCEG-L28 teaches a symbol mapping stage in which table selection is based on *previous_level*, i.e., the level value of a preceding run/level number pair (*the first number of a preceding number pair in the scanning order*). EX1003, ¶¶132-133.

¶

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).

5. Claim 3

Tsai teaches the additional limitations of this claim. FIG. 11 of Tsai (below) shows a “context-sensitive arithmetic coder 116” that “us[es] two different adaptive context models (1104, 1106).” Tsai, 16:23-26. Tsai thus discloses context models (i.e., *contexts*) of a context-based arithmetic coder. EX1003, ¶134.

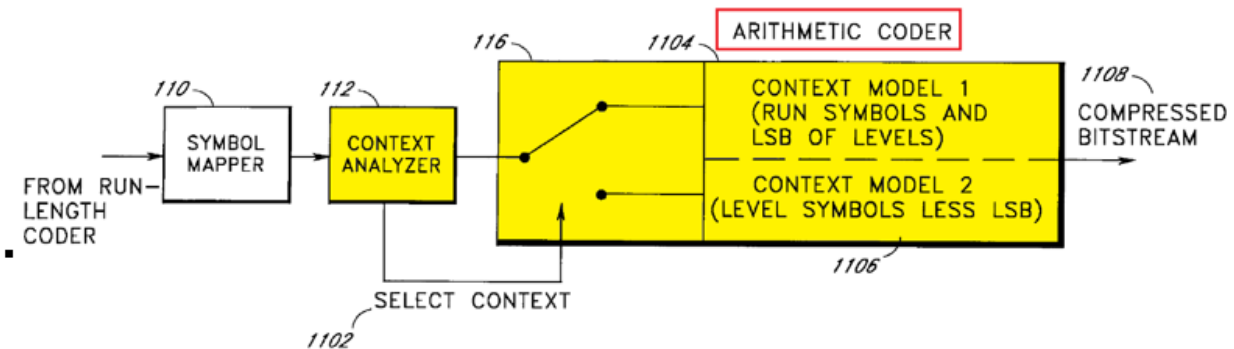


FIG. 11

Tsai, FIG. 11 (annotated).

6. Claim 4

Tsai teaches the additional limitations of this claim. As discussed for Claim 3, Tsai teaches a “context-sensitive arithmetic coder.” Tsai further 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 (*wherein said plurality of contexts are contexts of a context-based binary arithmetic coder*). EX1003, ¶135.

7. Claim 5

Tsai teaches the additional limitations of this claim. In the mapping of 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. 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 1 (*a first value*) and 0 (*a second value*). See EX1001, 12:46-49; EX1003, ¶136.

8. Claim 6

Tsai teaches the additional limitations of this claim. As shown in Tsai’s FIG. 7 (below), Symbol Map 1 and Symbol Map 2 are used to assign level values to a respective set of bins. Tsai, 11:45-51. For example, level value 2 is mapped to a set of bins assigned to the value of 1 (*the first value*). Tsai’s mapping would have been understood by a POSITA to be an example of known binarization technique that maps symbols to binary values (bins). EX1001, 12:46-49; EX1003, ¶137.

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).

Tsai thus teaches, for example, a 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*). EX1003, ¶137.

9. Claim 7

Tsai teaches the additional limitations of this claim. As discussed in 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). EX1003, ¶138.

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).

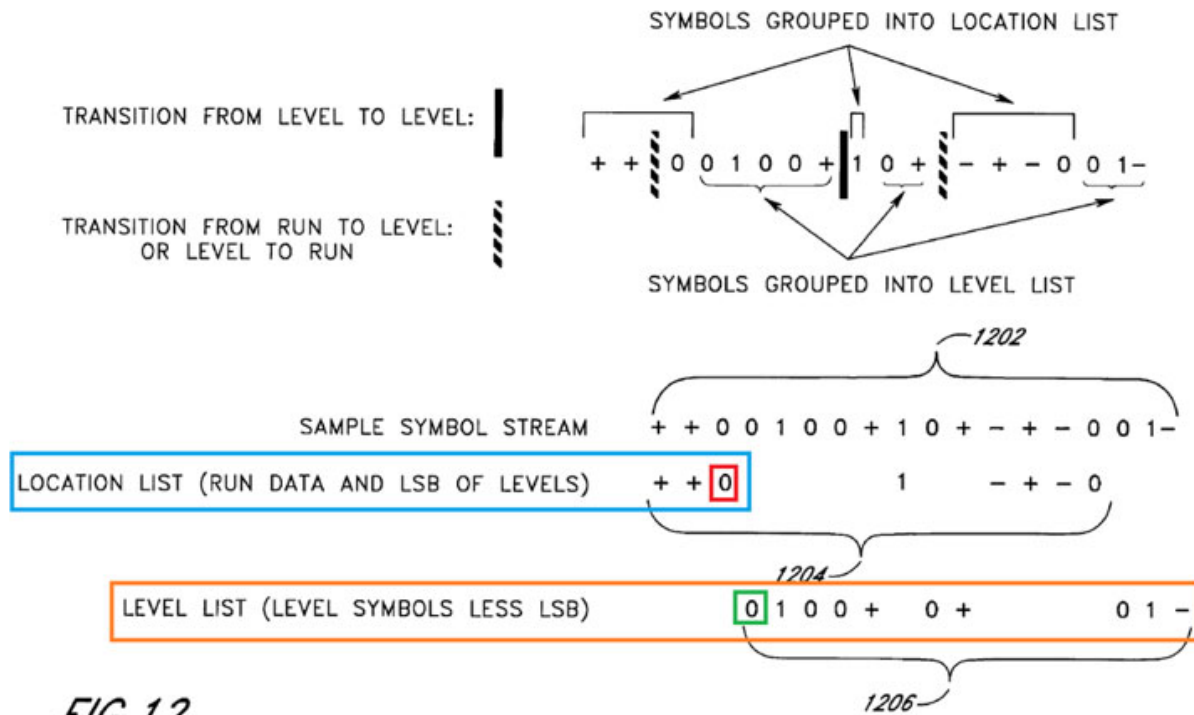
10. Claim 8

Tsai teaches 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. Tsai, 14:39-43; EX1003, ¶139.

<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).

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; EX1003, ¶140. 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.



Tsai, FIG. 12 (annotated).

Tsai thus teaches 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*). EX1003, ¶141.

11. Claim 9

Tsai teaches the additional limitations of this claim. Tsai discloses that the list 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. That is, the level value (*said first number*)

indicates the value or level of the nonzero coefficient (*is indicative of a non-zero coefficient value*), and the run value (*said second number*) indicates 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*). EX1003, ¶¶142-143.

12. Claim 10

a. 10[pre]

If limiting, Tsai teaches 10[pre] for the same reasons discussed for Ground 1A, 1[pre]. EX1003, ¶144.

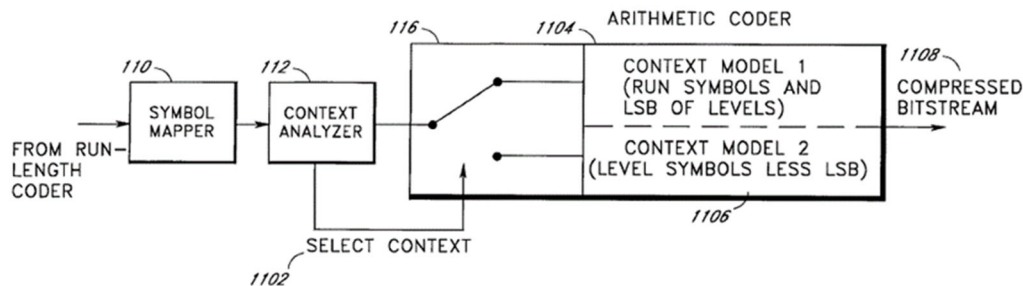
b. 10[a]

Tsai teaches 10[a] for the same reasons discussed for Ground 1A, 1[a]. EX1003, ¶145.

c. 10[b]

Tsai teaches this limitation. As discussed for Ground 1A, 1[b], Tsai teaches *a plurality of contexts*: a level context representative of the level words (the first numbers) and a run context representative of the run words. Furthermore, Tsai's FIG. 11 (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 second numbers (the run words in the location/run group) are assigned to the run context. EX1003, ¶147.



Tsai, FIG. 11.

Tsai therefore teaches 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*). EX1003, ¶148.

FIG. 12 of Tsai (below) illustrates context analyzer 112 partitioning the input symbol stream 1202 into two different types of symbol groups, where run values (*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.

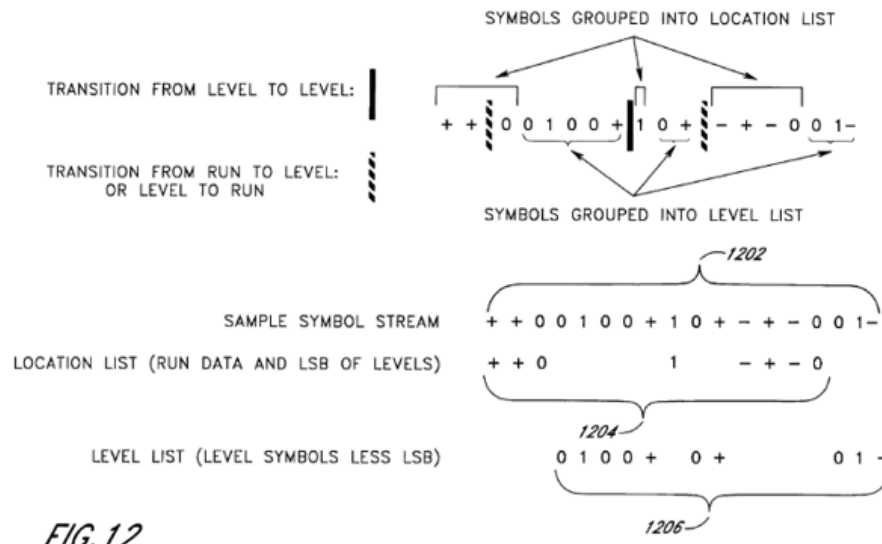


FIG. 12

Tsai, FIG. 12.

Tsai explains that instead of ending the symbol stream after the run word, the context analyzer 112 adds the LSB of the next level word to the run word before passing it to the arithmetic coder 116. Tsai, 15:25-30. For example, a run word “++” and level word “00100+” are concatenated as “new symbol group ‘++0’.” Tsai, 15:26-37. Context analyzer 112 then uses the coding context model 1104 to code these groups of symbols. Tsai, 16:7-11. Therefore, the context assigned to a run value is based on the partition that includes the LSB of the level word. EX1003, ¶¶149-151.

Tsai thus teaches that the run word (*second number of a number pair*) is assigned to a context at least partly in dependence on the LSB of the following level word in the pair (*first number of the number pair*). EX1003, ¶152.

13. Claim 11

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 3. EX1003, ¶153.

14. Claim 12

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 4. EX1003, ¶154.

15. Claim 13

Tsai teaches the additional limitations of this claim. Tsai teaches 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*). EX1003, ¶155.

16. Claim 14

Tsai teaches the additional limitations of this claim. As shown in Tsai’s FIG. 5, 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*). EX1003, ¶156. As explained for Claim 6, 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; EX1003, ¶¶156-157.

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.

17. Claim 17

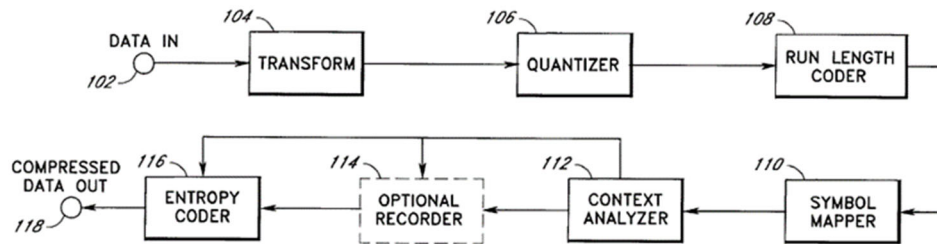
Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 9. EX1003, ¶158.

18. Claim 18

a. 18[pre]

If limiting, Tsai teaches 18[pre]. Tsai discloses coding methods in which a list of integers is used to represent quantized data. Tsai, 2:39-40. Tsai's FIG. 1 (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. Tsai, 6:61-64.

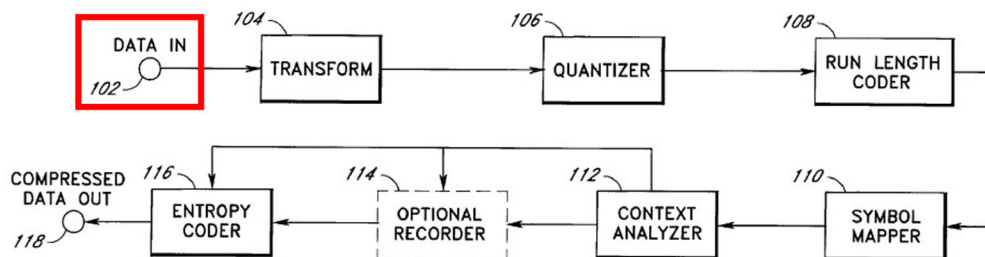


Tsai, FIG. 1.

Tsai therefore discloses a system for image coding. EX1003, ¶¶159-160.

b. 18[a]

Tsai teaches 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; EX1003, ¶161.

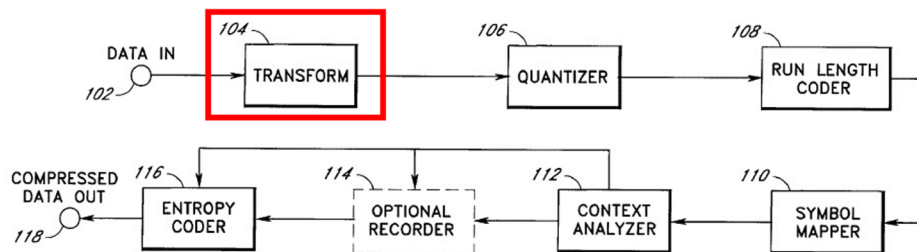


Tsai, FIG. 1 (annotated).

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*). EX1003, ¶162.

c. 18[b]

Tsai teaches 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 transform processor 104 (*a transform coder*), which uses known transforms, including the DCT, 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).

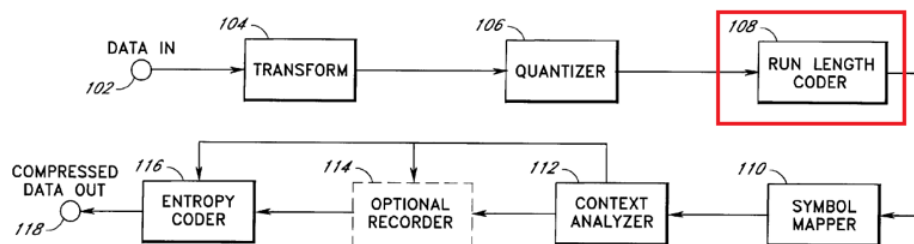
As shown in FIG. 4A of Tsai, the transformed data values are represented as a two-dimensional array of coefficients (*a corresponding block of transform coefficient values*). EX1003, ¶¶163-164.



Tsai, FIG. 4A.

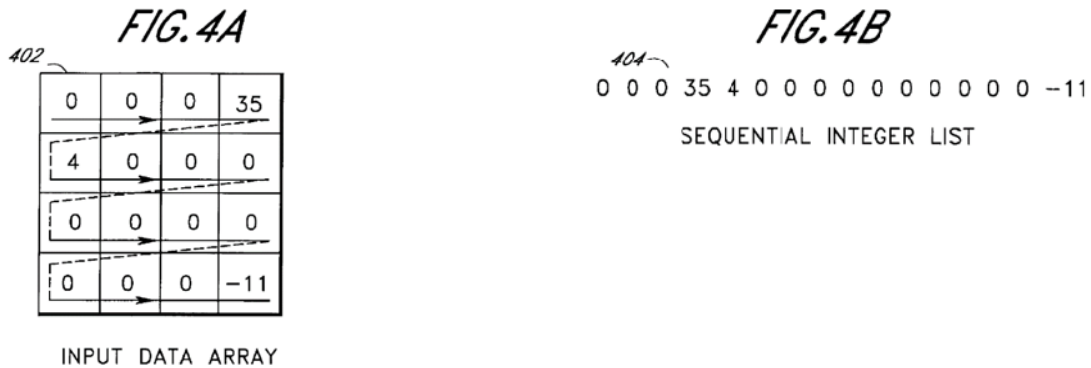
d. 18[c]

Tsai teaches 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*). Tsai, 9:30-34.



Tsai, FIG. 1 (annotated).

As shown in FIGS. 4A-B, the array of coefficient values is arranged according to a given order (*a scanned array of coefficient values, arranged according to the scanning order*). EX1003, ¶166.

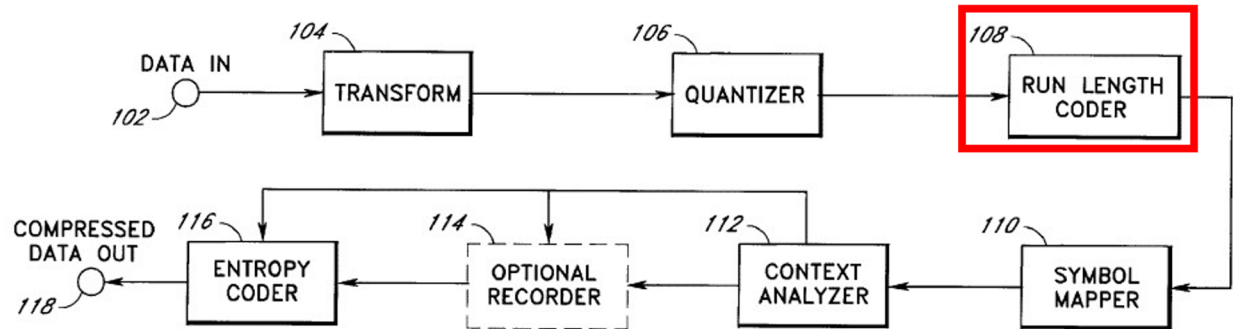


Tsai, FIGS. 4A-B.

e. 18[d]

Tsai teaches 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. EX1003, ¶167. Accordingly, 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

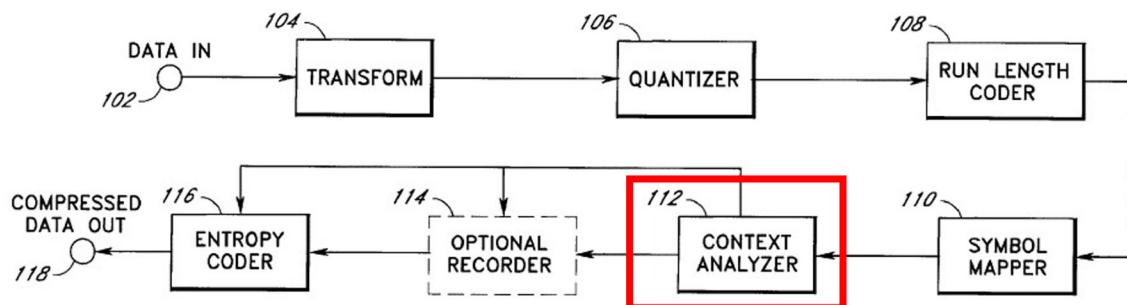
value (*a first number*) and a run value (*a second number*). Tsai, 10:5-9; EX1003, ¶167.



Tsai, FIG. 1 (annotated).

f. 18[e]

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 teaches the remaining features of 18[e] for the same reasons discussed for Ground 1A, 1[b]. EX1003, ¶168.



Tsai, FIG. 1 (annotated).

19. Claim 19

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

3. EX1003, ¶169.

20. Claim 20

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

4. EX1003, ¶170.

21. Claim 21

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

5. EX1003, ¶171.

22. Claim 22

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

6. EX1003, ¶172.

23. Claim 23

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

7. EX1003, ¶173.

24. Claim 24

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

8. EX1003, ¶174.

25. Claim 25

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim

2. EX1003, ¶175.

26. Claim 26

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 9. EX1003, ¶176.

27. Claim 27

a. 27[pre]

Tsai teaches 27[pre] for the same reasons discussed for Ground 1A, Claim 18[pre]. EX1003, ¶177.

b. 27[a]

Tsai teaches 27[a] for the same reasons discussed for Ground 1A, Claim 18[a]. EX1003, ¶178.

c. 27[b]

Tsai teaches 27[b] for the same reasons discussed for Ground 1A, Claim 18[b]. EX1003, ¶179.

d. 27[c]

Tsai teaches 27[c] for the same reasons discussed for Ground 1A, Claim 18[c]. EX1003, ¶180.

e. 27[d]

Tsai teaches 27[d] for the same reasons discussed for Ground 1A, Claim 18[d]. EX1003, ¶181.

f. 27[e]

Tsai teaches 27[e] for the same reasons discussed for Ground 1A, Claim 18[e]. EX1003, ¶182.

28. Claim 28

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 11. EX1003, ¶183.

29. Claim 29

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 12. EX1003, ¶184.

30. Claim 30

Tsai teaches Claim 30 for the same reasons discussed for Ground 1A, Claim 13. EX1003, ¶185.

31. Claim 31

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 14. EX1003, ¶186.

32. Claim 34

Tsai teaches this claim for the same reasons discussed for Ground 1A, Claim 9. EX1003, ¶187.

33. Claim 35

a. 35[pre]

Tsai teaches 35[pre] for the same reasons discussed for Ground 1A, 18[pre].
EX1003, ¶188.

b. 35[a]

Tsai teaches 35[a] for the same reasons discussed for Ground 1A, 18[a].
EX1003, ¶189.

c. 35[b]

Tsai teaches 35[b] for the same reasons discussed for Ground 1A, 18[b].
EX1003, ¶190.

d. 35[c]

Tsai teaches 35[c] for the same reasons discussed for Ground 1A, 18[c].
EX1003, ¶191.

e. 35[d]

Tsai teaches 35[d] for the same reasons discussed for Ground 1A, 18[d].
EX1003, ¶192.

f. 35[e]

Tsai teaches 35[e] for the same reasons discussed for Ground 1A, 18[e].
EX1003, ¶193.

34. Claim 36

a. 36[pre]

Tsai teaches 36[pre] for the same reasons discussed for Ground 1A, 27[pre].
EX1003, ¶194.

b. 36[a]

Tsai teaches 36[a] for the same reasons discussed for Ground 1A, 27[a].
EX1003, ¶195.

c. 36[b]

Tsai teaches 36[b] for the same reasons discussed for Ground 1A, 27[b].
EX1003, ¶196.

d. 36[c]

Tsai teaches 36[c] for the same reasons discussed for Ground 1A, 27[c].
EX1003, ¶197.

e. 36[d]

Tsai teaches 36[d] for the same reasons discussed for Ground 1A, 27[d].
EX1003, ¶198.

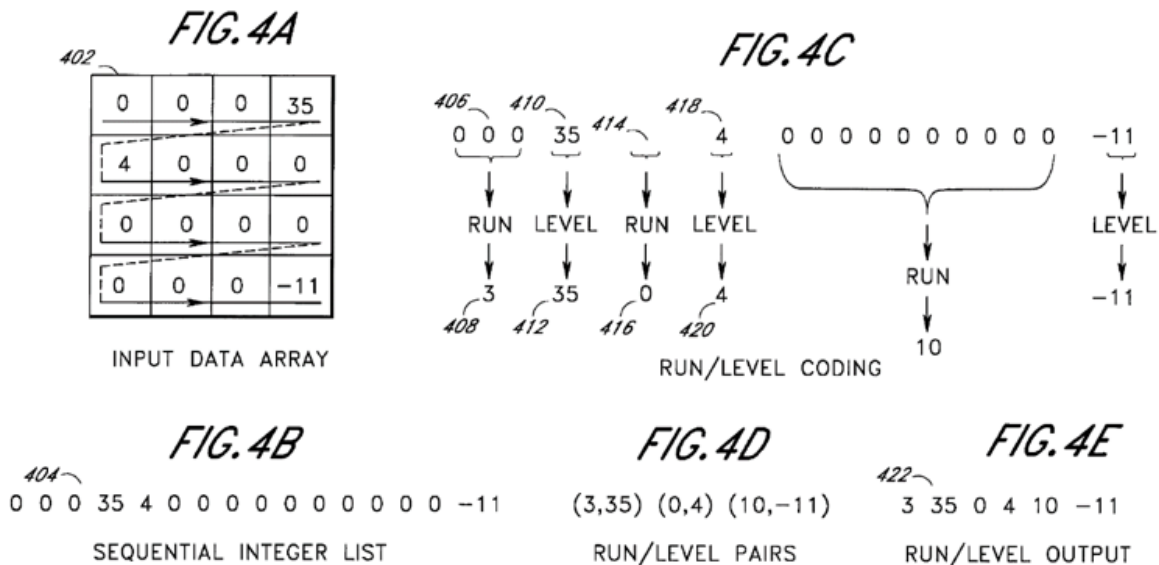
f. 36[e]

Tsai teaches 36[e] for the same reasons discussed for Ground 1A, 27[e].
EX1003, ¶199.

35. Claim 37

a. 37[pre]

To this extent this preamble is limiting, Tsai teaches it. For example, Tsai discloses an entropy coder 116 that is an adaptive arithmetic coder to perform context-based arithmetic encoding. FIGS. 4A-E of Tsai (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*). EX1003, ¶200.



Tsai, FIGS. 4A-E.

b. 37[a]

Tsai teaches 37[a] for the same reasons discussed for Ground 1A, 1[a]. EX1003, ¶201.

c. 37[b]

Tsai teaches 37[b] for the same reasons discussed for Ground 1A, 1[b].
EX1003, ¶202.

36. Claim 38

a. 38[pre]

As discussed in limitation 37[pre], Tsai discloses that an array of integer values (*an array of data symbols*) can be represented as a codeword output (*a code-word*). EX1003, ¶203.

b. 38[a]

Tsai teaches 38[a] for the same reasons discussed for Ground 1A, 10[a].
EX1003, ¶204.

c. 38[b]

Tsai teaches 38[b] for the same reasons discussed for Ground 1A, 10[b].
EX1003, ¶205.

37. Claim 39

a. 39[pre]

Tsai teaches 39[pre] for the same reasons discussed for Ground 1A, 37[pre].
EX1003, ¶206.

b. 39[a]

Tsai teaches 39[a] for the same reasons discussed for Ground 1A, 18[d].
EX1003, ¶207.

c. 39[b]

Tsai teaches 39[b] for the same reasons discussed for Ground 1A, 18[e].
EX1003, ¶208.

38. Claim 40

a. 40[pre]

Tsai teaches 40[pre] for the same reasons discussed for Ground 1A, 37[pre].
EX1003, ¶209.

b. 40[a]

Tsai teaches 40[a] for the same reasons discussed for Ground 1A, 27[d].
EX1003, ¶210.

c. 40[b]

Tsai teaches 40[b] for the same reasons discussed for Ground 1A, 27[e].
EX1003, ¶211.

**B. Ground 1B: Obviousness over Tsai, VCEG-L28, VCEG-L13
(Claims 15-16, 32-33)**

1. VCEG-L13 Overview

VCEG-L13 is a VCEG proposal for modifying the H.26L standard that indicates on its face that it was generated on January 4, 2001. VCEG-L13, 1. Dr. Wenger confirms that VCEG-L13 was submitted during the same January 9-12, 2001 Eibsee meeting as VCEG-L28. *See supra* §VI.A.2; Wenger, ¶16. VCEG-

L13 was made available on a public VCEG FTP server by January 8, 2001. Wenger, ¶¶18-38. VCEG-L13 is thus §102(a) prior art to the '701 patent.

VCEG-L13 recognizes that “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 table are not designed according to the symbol probabilities. VCEG-L13, 1; EX1003, ¶213. VCEG-L13 proposes the adoption of CABAC to address the issue. VCEG-L13, 1-2.

VCEG-L13 discloses context models for run/level pairs. VCEG-L13, 4-5. “A non-binary valued symbol will be decomposed into a sequence of binary decisions.” VCEG-L13, 5. Specifically, run and level data are each represented by multiple bins, each having a value of “0” or “1.” VCEG-L13, 5; EX1003, ¶214.

2. Claim 15

Tsai in view of VCEG-L28 and VCEG-L13 teaches the additional limitations of this claim.

As discussed in Ground 1A, 1[b], a POSITA implementing Tsai’s 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. In VCEG-L13, “RUN...[is] coded separately” in bins. VCEG-L13, 4. As shown in Table 2 (below), for all symbols, each bin of a run value is assigned a value of 1 (*the first value*) or a value of 0 (*the second value*). EX1003, ¶¶215-217.

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)

VCEG-L13, Table 2.

Combination of Tsai, VCEG-L28, and VCEG-L13: A POSITA implementing the Tsai-VCEG-L28 combination as described in Ground 1A would have been motivated to further incorporate VCEG-L13's CABAC coding techniques. EX1003, ¶218. Tsai emphasizes the importance of mapping symbols in a way that allows for unequal probability distributions to “lead to greater efficiency in the entropy coding.” EX1005, 10:42-45. VCEG-L13 provides a method where non-binary symbols are first converted into binary form, and then each binary decision is encoded using a binary arithmetic coding engine, guided by a probability model generated through context modeling. VCEG-L13, 2. This context model is continually updated with the data from previously encoded symbols, allowing the system to adapt quickly to changing statistical patterns, thereby improving coding efficiency. VCEG-L13, 2.

Specifically, VCEG-L13 describes the handling of run values in a way that employs different models: one for the initial part of the binary sequence and another for the remaining sequence. VCEG-L13, §3.3.2. A POSITA would have recognized that integrating VCEG-L13's adaptive context modeling, which uses models adaptive to the probability statistics of run values, would have aligned well with Tsai's common objective of enhancing coding efficiency through context-adaptive techniques. VCEG-L13, §3.3.2. By applying VCEG-L13's approach to Tsai's system, the overall coding performance would be improved, particularly in terms of efficiency and adaptability, achieving a more optimized entropy coding process. EX1003, ¶218.

Further, a POSITA would reasonably expected 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 algorithm in VCEG-L13 is an example of the adaptive arithmetic coding techniques disclosed in Tsai, making their teachings consistent. EX1003, ¶219.

3. Claim 16

Tsai in view of VCEG-L28 and VCEG-L13 teaches the additional limitations of this claim. VCEG-L13 teaches “[f]or 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, §3.3.2. 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*). EX1003, ¶¶220-222.

4. Claim 32

Tsai in view of VCEG-L28 and VCEG-L13 renders this claim obvious for the same reasons discussed for Ground 1B, Claim 15. EX1003, ¶223.

5. Claim 33

Tsai in view of VCEG-L28 and VCEG-L13 renders this claim obvious for the same reasons discussed for Ground 1B, Claim 16. EX1003, ¶224.

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

1. AAPA Overview

“A patentee’s admissions regarding the scope and content of the prior art [i.e., AAPA] under § 103 can be used, for example, to: (1) supply missing claim limitations that were generally known in the art prior to the invention...; (2)

support a motivation to combine particular disclosures; or (3) demonstrate the knowledge of the ordinarily-skilled artisan at the time of the invention...for any other purpose related to patentability.” *AAPA-Guidance*, 4 (citing *Qualcomm*, 24 F.4th at 1376; *Koninklijke Philips v. Google, LLC*, 948 F.3d 1330, 1337-38 (Fed. Cir. 2020)). As discussed *supra* §III.B.1, the ’701 patent Background section contains AAPA. *AAPA-Guidance*, 4-6.

2. Savatier Overview

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; EX1003, ¶¶226-232. 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.

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.

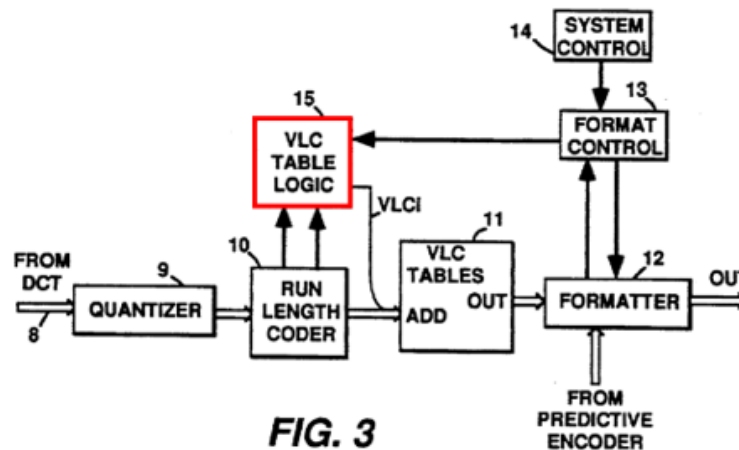


FIG. 3

Savatier, FIG. 3 (annotated).

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.

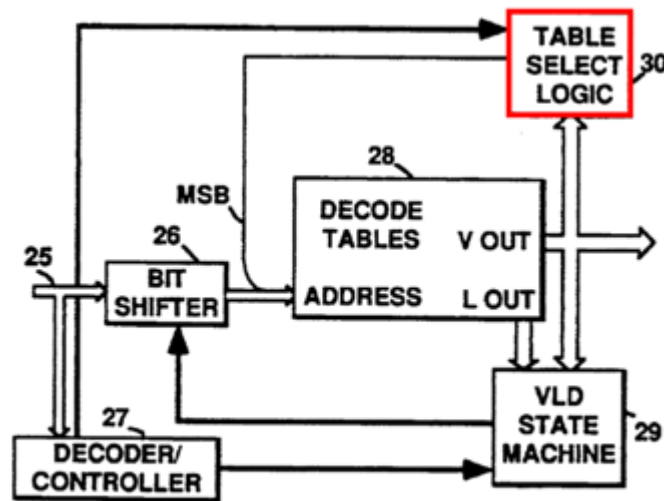


FIG. 4

Savatier, FIG. 4 (annotated).

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.

“The adaptivity of the variable length coding/decoding is dependent upon [*inter alia*]...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]

If limiting, AAPA discloses 1[pre]. AAPA discloses a method of video coding within the framework of the H.26L standard that a POSITA would have understood to be *a method of image coding*. EX1001, 11:36-47. 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; EX1003, ¶234.

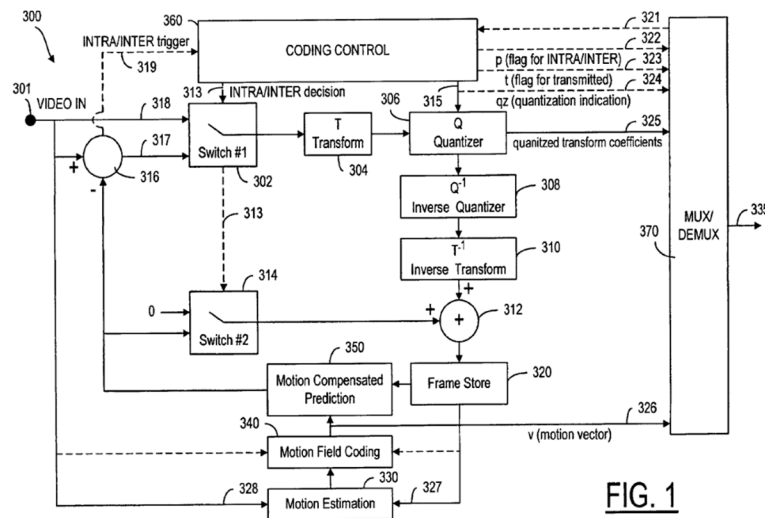


FIG. 1

EX1001, FIG. 1.

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.

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*). EX1003, ¶¶234-236.

AAPA’s “encoder 300...comprises a transformation unit 304 [that]...perform[s] a block-based ...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 [DCT] 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

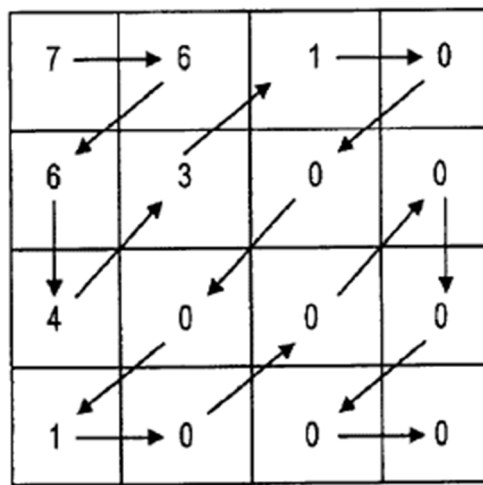
coefficients (*to produce a corresponding block of transform coefficient values*). EX1003, ¶237.

AAPA discloses this limitation. AAPA 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. 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*). EX1003, ¶238.

b. 1[a]

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 coefficients preceding the coefficient in question.” EX1001, 6:7-12.

Specifically, AAPA describes that the CABAC method used in H.26L encodes run-level pairs derived from the quantized DCT coefficients. EX1001, 14:40-65. FIG. 6a of AAPA shows a two-dimensional array of quantized DCT coefficient values that is zigzag-scanned to produce a one-dimensional array. EX1001, 14:40-53, FIG. 6a; EX1003, ¶239.



EX1001, FIG. 6a.

“The non-zero coefficient values in the one-dimensional array are then represented as pairs of run and level values” (*a plurality of number pairs*), and “each level value represents the value of a non-zero quantized DCT coefficient, while the associated run value corresponds to the number of zero-valued coefficients preceding the coefficient.” EX1001, 14:54-59; FIG. 6b.

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

↑ Run
↑ Level

FIG. 6b

EX1001, FIG. 6b.

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 having a level (*a first number*) and a run (*a second number*). EX1003, ¶¶239-241.

c. 1[b]

AAPA in view of Savatier teaches this limitation. AAPA discloses the CABAC method in which 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; EX1003, ¶243.

FIG. 6b

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

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; TABLE 1 (annotated).

Each of the bins of the level number is then “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 id.*, 15:3-15, FIG. 6c. Once again using the **(6, 0)** pair in FIG. 6b as an example, the sequence of bins representing the level number, **00000010** (red), is assigned to

Context1, Context2, and Context3, as illustrated in annotated Figure 6c and Table 1 below. EX1003, ¶244.

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

number

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

Level Run Level Run Level Run Level Run Level Run Level Run

↑

EOB

FIG. 6c

EX1001, Table 1 and FIG. 6C (annotated).

Specifically, 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; *see id.*, 23:3-11 (explaining the four contexts specified in H.26L TML8); EX1003, ¶245.

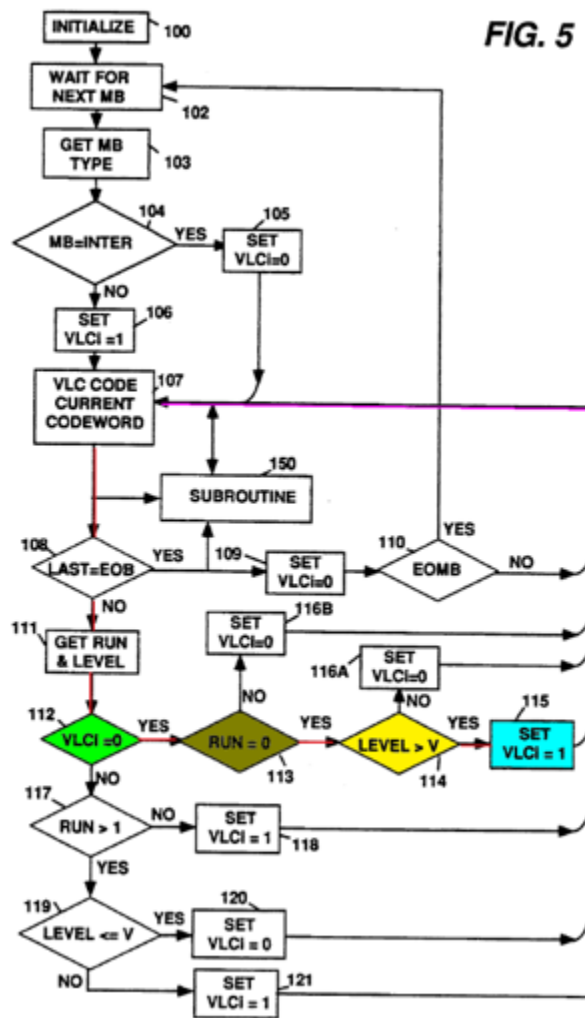
Thus, the CABAC scheme of H.26L described in AAPA discloses assigning level values (*the first numbers*) to at least one of Context1, Context2, Context3 representing the level values (*one of a plurality of contexts representative of the first numbers*). EX1003, ¶246.

As discussed *supra* §VI.C.2, Savatier discloses adaptively coding run/level data based on prior data. Savatier, Abstract, 1:5-8, 4:32-34. 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.

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. As discussed *supra* §VI.C.2, Savatier’s table selection is context assignment because Savatier’s coding/decoding tables are preprogrammed with VLC codes associated with anticipated signal statistics. These tables are used to encode/decode each successive codeword based on the specific context, ensuring that the most appropriate table is selected for the current data, optimizing the efficiency of the process. EX1003, ¶¶247-248. “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. EX1003, ¶248.

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, FIG. 5 (annotated).

The VLC table selection for current codeword [107] is thus based on the value of LEVEL in previous codeword [114]. EX1003, ¶¶249-250.

As explained *supra* §VI.C.2, Savatier's table selection is context 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*).
EX1003, ¶251.

Combination of AAPA and Savatier: A POSITA would have implemented Savatier's adaptive table selection method in AAPA's H.26L system to improve coding efficiency. 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 would realize Savatier's stated benefits in that system.
EX1003, ¶252.

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; EX1003, ¶253. 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. Savatier, 5:33-40, 5:62-6:35, 8:9-20, FIG. 5.

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

↑ Run
Level

FIG. 6b

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).

In the combination according to Savatier's scheme, 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. EX1003, ¶¶253-254. 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.

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 for several reasons. EX1003, ¶255.

First, a POSITA would have recognized that Savatier and AAPA disclose complementary coding techniques within the field of entropy coding in video compression systems. Savatier, 1:64-68, 2:39-48, FIGS. 1-2; EX1001, 11:35-12:34; EX1003, ¶256.

Second, a POSITA would have recognized that Savatier improves 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 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 and higher efficiency. EX1003, ¶257.

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). EX1003, ¶258; *KSR*, 550 U.S. at 417.

4. Claim 2

AAPA in view of Savatier teaches the additional limitations of this claim. As discussed in Ground 2, 1[b], 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*). EX1003, ¶259.

5. Claim 3

AAPA teaches the additional limitations of this claim. 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; EX1003, ¶260.

6. Claim 4

AAPA teaches the additional limitations of this claim. 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; EX1003, ¶261.

7. Claim 5

AAPA teaches 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, such as level values in the level-run number pairs (*the first numbers of said number pairs*), to a sequence of **bins**, each of which can be assigned a value of either 1 (*a first value*) or 0 (*a second value*). EX1001, 12:42-50, Table 1 (annotated below for the bin sequence "0001" for value "3"); EX1003, ¶262.

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

AAPA teaches 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*). EX1003, ¶263

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

AAPA teaches the additional limitations of this claim. As shown in Table 1 of AAPA (below), the value of a level (e.g., 3) is represented using bins having values of 1 (*first value is 1*) and 0 (*second value is 0*). EX1001, 12:42-50; EX1003, ¶264.

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

AAPA teaches the additional limitations of this claim. In AAPA, the assignment of the context for a bin is based on the bin's position. For example, the first bin "0" in the **bin sequence** for value 3 is assigned to context 1 (green), the second bin "0" is assigned to context 2 (yellow), and the third bin "0" and fourth bin "1" are both assigned to context 3 (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*). EX1003, ¶266.

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									
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	0	0
Context 2	3	0	0	0	0	0	0	0	0	1
	2	0	0	0	0	0	0	0	0	1
Context 1	1	0	1	0	1	0	1	0	1	0

Level 7 0 Run

0 End

FIG. 6c

EX1001, Table 1 and FIG. 6C (annotated).

11. Claim 9

AAPA teaches the additional limitations of this claim. 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; EX1003, ¶267.

12. Claim 18

a. 18[pre]

If limiting, AAPA teaches 18[pre]. Figure 1 of AAPA (below) illustrates a prior art “video encoder that employs a combination of INTRA- and INTER-coding to produce a compressed (encoded) video bit-stream” (*a system for image coding*). EX1001, 4:27-30; EX1003, ¶268.

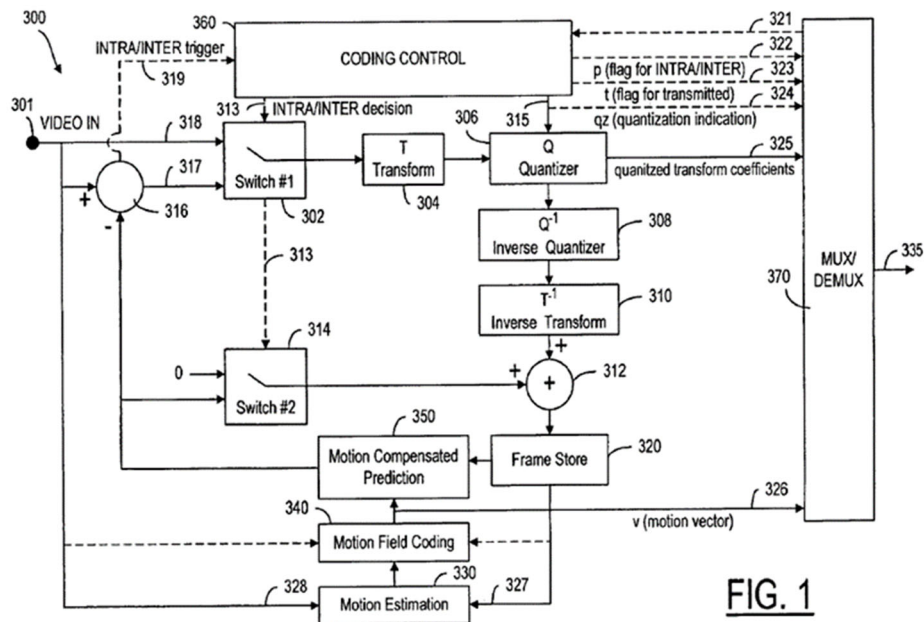


FIG. 1

EX1001, FIG. 1.

b. 18[a]

AAPA teaches this limitation. AAPA's encoder 300 comprises 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, 5:36-38, 1:21-23.

AAPA thus teaches 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*). EX1003, ¶¶269-270.

c. 18[b]

AAPA teaches this limitation. AAPA discloses a transformation unit 304 (*a transform coder*) that performs “a 2-dimensional [DCT] on each block of values, producing a 2-dimensional array of DCT coefficients for each block.” EX1001, 5:32-40; 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 [DCT] on each block of values, producing a 2-dimensional array of DCT coefficients for each block.” EX1001, 5:30-35. 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-dimensional array of DCT coefficients (*to produce a corresponding block of transform coefficient values*). EX1003, ¶¶271-272.

d. 18[c]

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*). EX1003, ¶273.

e. 18[d]

AAPA teaches this limitation. AAPA discloses that the video multiplex coder 370 performs the function of “represent[ing] each non-zero quantized coefficient in the one dimensional array by two values, referred to as level and run.” EX1001, 6:7-17; Ground 2, 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*). EX1003, ¶274.

f. 18[e]

AAPA in view of Savatier teaches this limitation. AAPA discloses a prior art context-based binary arithmetic encoder 700 (specifically, context assignment block 720 thereof) that assigns a symbol (level or run number) to a “corresponding context,” as shown in FIG. 4 (below). EX1001, 13:20-23; *see id.*, 12:65-13:44.

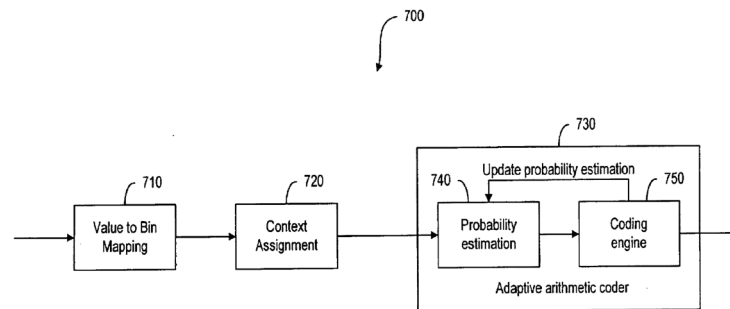


FIG. 4
(PRIOR ART)

EX1001, FIG. 4.

Specifically, Figure 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.” EX1001, 15:5-15, Table 1, FIG. 6c (annotated below); Ground 2, 1[b].

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

Level → 7
 ← Run 6
 0
 EOB

FIG. 6c

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

A POSITA would have understood that encoder 700 assigns a bin sequence converted from a level (*first*) number to contexts 1, 2, and 3 according to the process in H.26L described in AAPA. EX1003, ¶277.

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; EX1003, ¶278.

AAPA's encoder 700 assigns a bin sequence converted from a level (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]. EX1003, ¶279.

13. Claim 19

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 3. EX1003, ¶280.

14. Claim 20

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 4. EX1003, ¶281.

15. Claim 21

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 5. EX1003, ¶282.

16. Claim 22

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 6. EX1003, ¶283.

17. Claim 23

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 7. EX1003, ¶284.

18. Claim 24

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 8. EX1003, ¶285.

19. Claim 25

The AAPA-Savatier combination renders Claim 25 obvious for the same reasons discussed for Ground 2, Claim 2. EX1003, ¶286.

20. Claim 26

AAPA teaches Claim 26 for the same reasons discussed for Ground 2, Claim 9. EX1003, ¶287.

21. Claim 35

a. 35[pre]

AAPA discloses 35[pre] for the same reasons discussed for Ground 2, 18[pre]. EX1003, ¶288.

b. 35[a]

AAPA discloses 35[a] for the same reasons discussed for Ground 2, 18[a]. EX1003, ¶289.

c. 35[b]

AAPA discloses 35[b] for the same reasons discussed for Ground 2, 18[b]. EX1003, ¶290.

d. 35[c]

AAPA discloses 35[c] for the same reasons discussed for Ground 2, 18[c]. EX1003, ¶291.

e. 35[d]

AAPA discloses 35[d] for the same reasons discussed for Ground 2, 18[d]. EX1003, ¶292.

f. 35[e]

AAPA in view of Savatier discloses 35[e] for the same reasons discussed for Ground 2, 18[e]. EX1003, ¶293.

22. Claim 37

a. 37[pre]

If limiting, AAPA discloses 37[pre] because AAPA 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 (below); EX1003, ¶294.

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. 37[a]

AAPA discloses 37[a] for the same reasons discussed for Ground 2, 1[a]. EX1003, ¶295.

c. 37[b]

AAPA combined with Savatier teaches 37[b] for the same reasons discussed for Ground 2, 1[b]. EX1003, ¶296.

23. Claim 39

a. 39[pre]

If limiting, AAPA discloses 39[pre] because 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 “70” (*a code-word*). EX1001, 5:56-64, FIG. 3, 14:46-53, FIG. 6b (below); EX1003, ¶297.

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]

AAPA discloses 39[a] for the same reasons discussed for Ground 2, 18[d]. EX1003, ¶298.

c. 39[b]

AAPA combined with Savatier discloses 39[b] for the same reasons discussed for Ground 2, 18[e]. EX1003, ¶299.

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

1. Claim 10

a. 10[pre]

If limiting, AAPA discloses 10[pre] for the same reasons discussed above for Ground 2, 1[pre]. EX1003, ¶300.

b. 10[a]

AAPA discloses this limitation for the same reasons discussed above for Ground 2, 1[a]. EX1003, ¶301.

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.

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.

In VCEG-L13, the level value is first processed and assigned to context(s). VCEG-L13, §3.3.1. VCEG-L13 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*). EX1003, ¶303.

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

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 from 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. EX1003, ¶305.

Second, according to VCEG-L13, “[i]f $LEVEL \neq 0$ (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. EX1003, ¶306.

Third, the AAPA-VCEG-L13 combination yields 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 therefore have recognized that incorporating VCEG-L13's method into H.26L would enhance compression efficiency across various bit rates. EX1003, ¶307; VCEG-L13, §7.

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."); EX1003, ¶308.

2. Claim 11

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 3. EX1003, ¶309.

3. Claim 12

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 4. EX1003, ¶310.

4. Claim 13

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, Section 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, Section 3.3.2. Therefore, AAPA in view of VCEG-L13 teaches mapping of RUN values (*the second numbers of said number pairs*) to **bins**, each of which can be assigned a value of either 1 (*a first value*) or 0 (*a second value*). EX1003, ¶311.

5. Claim 14

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 6. EX1003, ¶312.

6. Claim 15

AAPA teaches this claim for the same reasons discussed for Ground 2, Claim 7. EX1003, ¶313.

7. Claim 16

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. Therefore, 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 the first bin or remaining bin (*at least partly in dependence on the bin to which the second number of the number pair is mapped*). EX1003, ¶314.

8. Claim 17

AAPA teaches this claim for the same reasons discussed for Claim 9. EX1003, ¶315.

9. Claim 27

a. 27[pre]

AAPA discloses 27[pre] for the same reasons discussed for Ground 2, 18[pre]. EX1003, ¶316.

b. 27[a]

AAPA discloses 27[a] for the same reasons discussed for Ground 2, 18[a]. EX1003, ¶317.

c. 27[b]

AAPA discloses 27[b] for the same reasons discussed for Ground 2, 18[b]. EX1003, ¶318.

d. 27[c]

AAPA discloses 27[c] for the same reasons discussed for Ground 2, 18[c]. EX1003, ¶319.

e. 27[d]

AAPA discloses 27[d] for the same reasons discussed for Ground 2, 18[d]. EX1003, ¶320.

f. 27[e]

AAPA combined with VCEG-L13 discloses 27[e] for the same reasons discussed for Ground 3, 10[b]. EX1003, ¶321.

10. Claim 28

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 11. EX1003, ¶322.

11. Claim 29

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 12. EX1003, ¶323.

12. Claim 30

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 13. EX1003, ¶324.

13. Claim 31

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 14. EX1003, ¶325.

14. Claim 32

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 15. EX1003, ¶326.

15. Claim 33

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 16. EX1003, ¶327.

16. Claim 34

AAPA discloses this claim for the same reasons discussed for Ground 3, Claim 17. EX1003, ¶328.

17. Claim 36

a. 36[pre]

AAPA discloses 36[pre] for the same reasons discussed for Ground 3, 27[pre]. EX1003, ¶329.

b. 36[a]

AAPA discloses 36[a] for the same reasons discussed for Ground 3, 27[a]. EX1003, ¶330.

c. 36[b]

AAPA discloses 36[b] for the same reasons discussed for Ground 3, 27[b]. EX1003, ¶331.

d. 36[c]

AAPA discloses 36[c] for the same reasons discussed for Ground 3, 27[c]. EX1003, ¶332.

e. 36[d]

AAPA discloses 36[d] for the same reasons discussed for Ground 3, 27[d]. EX1003, ¶333.

f. 36[e]

AAPA in view of Savatier discloses 36[e] for the same reasons discussed for Ground 3, 27[e]. EX1003, ¶334.

18. Claim 38

a. 38[pre]

As discussed for Ground 2, 37[pre], AAPA discloses 38[pre] because AAPA discloses 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; EX1003, ¶335.

b. 38[a]

AAPA discloses 38[a] for the same reasons discussed for Ground 3, 10[a]. EX1003, ¶336.

c. 38[b]

AAPA combined with VCEG-13 teaches 38[b] for the same reasons discussed for Ground 3, 10[b]. EX1003, ¶337.

19. Claim 40

a. 40[pre]

AAPA discloses 40[pre] for the same reasons discussed for Ground 2, 39[pre]. EX1003, ¶338.

b. 40[a]

AAPA discloses 40[a] for the same reasons discussed for Ground 2, 18[d]. EX1003, ¶339.

c. 40[b]

AAPA combined with VCEG-L13 discloses 40[b] for the same reasons discussed for Ground 3, 10[b]. EX1003, ¶340.

VII. DISCRETIONARY DENIAL WOULD BE INAPPROPRIATE

As far as Petitioner is aware, this is the first time that the '701 patent has been challenged in a post-grant proceeding. There is one proceeding involving the '701 patent pending before the district court in *Nokia Techs. Oy v. Amazon.com, Inc.*, initiated on October 31, 2023. No schedule has been entered. The latest statistics indicate that the median time from filing to trial in the District of Delaware is 32.9 months. EX1023, 14. Therefore, the expected trial date is July 27, 2026, after the likely FWD deadline of March 2026. Further, it is unclear whether any overlap yet exists between the Petition and issues that might be raised in the district court. Discretionary denial under *Fintiv* is thus unwarranted.

Denial under §325(d) is likewise unwarranted because Tsai, VCEG-L28, Savatier, and VCEG-13 were not considered by the Examiner during prosecution.

VIII. CONCLUSION

IPR of the Challenged Claims is respectfully requested.

IX. MANDATORY NOTICES

A. Real Party in Interest

Petitioner identifies themselves as real parties in interest. Out of an abundance of caution, Petitioner discloses Twitch Interactive, Inc., as a potential real party in interest.

B. Related Matters

To the best of Petitioner's knowledge, the '701 patent has been involved in the following matters:

- *Nokia Techs. Oy v. Amazon.com, Inc.*, Case No. 1:23-cv-01236 (D. Del. Oct. 31, 2023)
- *Nokia Techs. Oy v. HP Inc.*, No. 1:23-cv-01237 (D. Del. Oct. 31, 2023)
- *Nokia Corp. v. Amazon.com, Inc.*, No. 1:23-cv-01232 (D. Del. Oct. 27, 2023)
- *Nokia Techs. Oy v. Lenovo (Shanghai) Elecs. Tech. Co.*, No. 5:19-cv-00427 (E.D.N.C. Sept. 25, 2019)
- *Nokia Techs. Oy v. Apple Inc.*, No. 2:16-cv-01440 (E.D. Tex. Dec. 21, 2016)
- *Certain Electronic Devices, Including Computers, Tablet Computers, and Components and Modules Thereof*, Inv. No. 337-TA-1208 (I.T.C. July 2, 2020)

- *Certain Video Capable Electronic Devices, Including Computers, Streaming Devices, Televisions, and Components and Modules Thereof*, Inv. No. 337-TA-1380 (I.T.C. Oct. 31, 2023)

C. Notice of Counsel and Service Information

LEAD COUNSEL
Jessica Kaiser (Reg. No. 58,937) kaiser-ptab@perkinscoie.com PERKINS COIE LLP 1900 Sixteenth Street, Suite 1400 Denver, CO 80202-5255 Telephone: (303) 291-2300
BACK-UP COUNSEL
Robin Brewer, Reg. No. 60,951 PERKINS COIE LLP 505 Howard Street Suite 1000 San Francisco, CA 94105 Tel: (415) 344-7000 Brewer-ptab@perkinscoie.com Atanas Baitchev, Reg No. 82,414 PERKINS COIE LLP 11452 El Camino Real Suite 300 San Diego, CA 92130-2080 Tel: (858) 720-5700 baitchev_ptab@perkinscoie.com Wei Yuan, Reg No. 71,772

PERKINS COIE LLP
11452 El Camino Real
Suite 300
San Diego, CA 92130-2080
Tel: (858) 720-5700
wyuan@perkinscoie.com

Ji Wang, Reg No. 80,018
PERKINS COIE LLP
11452 El Camino Real
Suite 300
San Diego, CA 92130-2080
Tel: (858) 720-5700
jiwang@perkinscoie.com

Jingyuan Huang, Reg. No. 74,962
PERKINS COIE LLP
3150 Porter Dr.
Palo Alto, CA 94304-1212
Tel: (650) 838-4300
Huang-ptab@perkinscoie.com

Jon Carter, Reg. No. 75,145
PERKINS COIE LLP
1155 Avenue of the Americas 22nd Floor.
New York, NY 10036-2711
Tel: (212) 262-6900
JCarter@perkinscoie.com

Amazon-NokiaIPR@perkinscoie.com

Respectfully submitted,

/ Jessica Kaiser /

Jessica Kaiser
Reg. No. 58,937
Attorney for Petitioner

PERKINS COIE LLP
1900 Sixteenth Street, Suite 1400
Denver, CO 80202-5255

Date: September 16, 2024

CERTIFICATE OF WORD COUNT UNDER 37 CFR §42.24(d)

Pursuant to 37 C.F.R. §42.24(a), Petitioner hereby certifies that portions of the above-captioned Petition for *inter partes* review of U.S. Patent No. 6,856,701, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 13,991. Pursuant to 37 C.F.R. §42.24(a), this word count is in compliance and excludes the table of contents, table of authorities, mandatory notices under §42.8, certificate of service, certificate of word count, appendix of exhibits, and any claim listing. This word count was prepared using Microsoft Word.

Respectfully submitted,

/ *Jessica Kaiser* /

Jessica Kaiser
Reg. No. 58,937
Attorney for Petitioner

Date: September 16, 2024
PERKINS COIE LLP
1900 Sixteenth Street, Suite 1400
Denver, CO 80202-5255

CERTIFICATE OF SERVICE

The undersigned hereby certifies that true copies of the Petition for *inter partes* review of U.S. Patent No. 6,856,701 and supporting materials (Exhibits and Power of Attorney) were served via overnight delivery on the Patent Owner at the correspondence address of record as listed on PAIR:

WARE, FRESSOLA, MAGUIRE & BARBER LLP
BRADFORD GREEN, BUILDING 5
755 MAIN STREET
MONROE, CT 06468

A courtesy copy was also sent via electronic mail to Patent Owner's litigation counsel listed below:

Brian E. Farnan
bfarnan@farnanlaw.com
Michael J. Farnan
mfarnan@farnanlaw.com
Warren H. Lipschitz
wlipschitz@mckoolsmith.com
Alexandra F. Easley
aeasley@mckoolsmith.com
Joshua Budwin
jbudwin@mckoolsmith.com
R. Mitch Verboncoeur
mverboncoeur@mckoolsmith.com
Josh Newcomer
jnewcomer@mckoolsmith.com
Kevin Burgess
kburgess@mckoolsmith.com
Theodore Stevenson, III
ted.stevenson@alston.com
John D. Haynes
john.haynes@alston.com

Nicholas T. Tsui
nick.tsui@alston.com
Shawn Gannon
shawn.gannon@alston.com

Respectfully submitted,

/ Jessica Kaiser /

Jessica Kaiser
Reg. No. 58,937
Attorney for Petitioner

Date: September 16, 2024