

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.

IPR2024-01141

U.S. Patent No. 8,036,273

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 8,036,273**

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

I.	INTRODUCTION	1
II.	GROUND FOR STANDING.....	1
III.	IDENTIFICATION OF CHALLENGE	1
	A. Prior Art.....	1
	B. Grounds	1
IV.	'273 PATENT OVERVIEW	2
	A. Technology Background	2
	B. The Alleged Problem in the Prior Art	7
	C. The Alleged Invention of the '273 Patent.....	9
V.	LEVEL OF ORDINARY SKILL	10
VI.	CLAIM CONSTRUCTION	11
VII.	PRIOR ART OVERVIEW	12
	A. TML6.....	12
	B. Fandrianto.....	17
VIII.	DETAILED EXPLANATION OF GROUNDs	21
	A. Ground 1: Obviousness over TML6 and Fandrianto (Claims 1, 4-17, 19-26, 28-29, 31, 33, 36-49, 51, 53-59, 61-62, 64-78, and 82-83)	21
	1. Motivation to Combine	21
	2. Claims 1, 4-17, 19-26, and 28-29	29
	3. Independent Claim 31	111
	4. Claims 33, 36-49, 51, 53-59, and 61-62.....	111
	5. Claims 64-67	113
	6. Claims 68-70	115
	7. Claims 71-72	117
	8. Claims 73-74	117
	9. Independent Claim 75	117
	10. Claims 76-78	118

11.	Independent Claim 82	119
12.	Independent Claim 83	119
B.	Ground 2: Claims 32, 52, and 79-81 are Obvious over TML6 in View of Fandrianto and the Knowledge of a POSITA	120
1.	Independent Claim 32	120
2.	Claim 52	122
3.	Claims 79-81	122
IX.	NO BASIS FOR DISCRETIONARY INSTITUTION DENIAL	123
A.	<i>General Plastic</i>	123
B.	<i>Fintiv</i>	125
C.	§325(d)	125
X.	CONCLUSION	127
XI.	MANDATORY NOTICES	127
A.	Real Party in Interest	127
B.	Related Matters	127
C.	Notice of Counsel and Service Information	128

EXHIBIT LIST

Exhibit	Reference
1001	U.S. Patent No. 8,036,273 (“the ’273 patent”)
1002	File History of the ’273 patent
1003	Declaration of Dr. Kannan Ramchandran (“Ramchandran”)
1004	Curriculum Vitae of Dr. Kannan Ramchandran
1005	H.26L Test Model Long Term Number 6 (TML-6) draft0 Reference Code (“TML6”)
1006	U.S. Patent No. 5,594,813 (“Fandrianto”)
1007	VCEG-L45 H.26L Test Model Long Term Number 6 (TML-6) draft0 (“VCEG-L45”)
1008	Declaration of Dr. Stephan Wenger (“Wenger”)
1009	U.S. Patent No. 6,295,376 (“Nakaya”)
1010	Original Complaint, <i>Nokia Techs. Oy v. Amazon.com, Inc.</i> , No. 1:23-cv-01236 (D. Del. Oct. 31, 2023), ECF No. 1.
1011	Martyn J. Riley and Iain E.G. Richardson, <i>Digital Video Communications</i> , Artech House, Inc. (1997) (“Riley”)
1012	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”)
1013	Ushashree, <i>Video Compression Based on RMB</i> , International Journal of Electronics Communication and Computer Engineering (2013) (“Ushashree”)
1014	Iain Richardson, <i>Video Codec Design: Developing Image and Video Compression Systems</i> (2002) (“Richardson”)
1015	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”)
1016	“Transfer of ftp://standards.pictel.com to ITU,” https://www.itu.int/wftp3/av-

Exhibit	Reference
	arch/Transfer%20of%20standards.pictel.com%20to%20ITU.htm (2003)
1017	ITU, https://www.itu.int/wftp3/av-arch/video-site/h26L/older_tml/
1018	Electrical Engineering Dictionary CRCnetBASE 2000, <i>codec</i> (2000) (“CRC-Dictionary”)
1019	Original Complaint, <i>Nokia Techs. Oy v. Lenovo (Shanghai) Elecs. Tech. Co.</i> , No. 5:19-cv-00427 (E.D.N.C. Sept. 25, 2019), ECF No. 1
1020	Joint Stipulation to Dismissal Without Prejudice, <i>Nokia Techs. Oy v. Lenovo (Shanghai) Elecs. Tech. Co.</i> , No. 5:19-cv-00427 (E.D.N.C. Apr. 12, 2021), ECF No. 126
1021	Gisle Bjontegaard et al., A simplified 1/3 pixel prediction method (Mar. 21, 2001) (“VCEG-M28”)
1022	<i>Interpolation</i> , The New Encyclopædia Britannica (1986) (“Britannica”)
1023	VCEG-N10 H.26L Test Model Long Term Number 8 (TML-8) draft0 (“VCEG-N10”)
1024	William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, <i>Numerical Recipes in C: The Art of Scientific Computing</i> , Cambridge University Press (2 nd ed. 1992) (reprinted 1996) (“Press”)

LISTING OF CHALLENGED CLAIMS

Claim 1	
1[pre][a]	A method for sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
1[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the method comprising causing an apparatus to:
1[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
1[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
1[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a

	weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 4	
4	A method according to claim 1, comprising causing the apparatus to use a first and a second weight in the weighted average chosen to interpolate the sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, wherein when the chosen weighted average is that involving the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, causing the apparatus to select the relative magnitudes of the first and second weights to be inversely proportional to the respective straight-line diagonal distances of the sub-pixels having co-ordinates with even values of K and L to the sub-pixel having co-ordinates with odd values of K and L whose value is being interpolated.
Claim 5	
5	A method according to claim 4, comprising causing the apparatus to use first and second weights with equal values when the sub-pixels having co-ordinates with even values of K and L are equidistant from the sub-pixel having co-ordinates with odd values of K and L whose value is being interpolated.
Claim 6	
6	A method according to claim 1, comprising causing the apparatus to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero

	and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions when the value of a sub-pixel having co-ordinates with K equal to an even value and L equal to an odd value is also required.
Claim 7	
7	A method according to claim 1, comprising causing the apparatus to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions when the value of a sub-pixel having co-ordinates with K equal to an odd value and L equal to an even value is also required.
Claim 8	
8	A method according to claim 1, comprising causing the apparatus to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an odd value O and L equal to an even value E, not including zero or 2^N , by taking an average of the value of a first sub-pixel having co-ordinates with K equal to the even value O-1 and L equal to E and a second sub-pixel having co-ordinates with K equal to the even value O+1 and L equal to E.
Claim 9	
9	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 1 and L equal to zero, by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to zero, and a sub-pixel having co-ordinates with K equal to 2 and L equal to zero.
Claim 10	
10	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-

	ordinates with K equal to 2^{N-1} and L equal to zero, by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to zero, and a sub-pixel having co-ordinates with K equal to 2^{N-2} and L equal to zero.
Claim 11	
11	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 1 and L equal to 2^N , by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to 2 and L equal to 2^N .
Claim 12	
12	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^{N-1} and L equal to 2^N , by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to 2^{N-2} and L equal to 2^N .
Claim 13	
13	A method according to claim 1, comprising causing the apparatus to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value E, not including zero or 2^N , and L equal to an odd value O, by taking an average of the value of a first sub-pixel having co-ordinates with K equal to E and L equal to the even value O-1 and a second sub-pixel having co-ordinates with K equal to E and L equal to the even value O+1.
Claim 14	
14	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to zero and L equal to 1, by taking an average of the value of a pixel having co-ordinates with K equal

	to zero and L equal to zero, and a sub-pixel having co-ordinates with K equal to zero and L equal to 2.
Claim 15	
15	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^N and L equal to 1, by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to zero, and a sub-pixel having co-ordinates with K equal to 2^N and L equal to 2.
Claim 16	
16	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to zero and L equal to 2^N-1 , by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to zero and L equal to 2^N-2 .
Claim 17	
17	A method according to claim 1, comprising causing the apparatus to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^N and L equal to 2^N-1 , by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to 2^N and L equal to 2^N-2 .
Claim 19	
19	A method according to claim 1, in which N is one of the values 2, 3, and 4.
Claim 20	
20	A method according to claim 1, wherein the pixels have a predetermined dynamic range, said predetermined dynamic range corresponding to the range of values said pixels can take.

Claim 21	
21	A method according to claim 20, comprising interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and for sub-pixels having co-ordinates with K equal to zero and L equal to an even value, using weighted sums of the values of pixels located in rows and columns respectively, to produce a first intermediate value, said first intermediate value having a dynamic range equal to the predetermined dynamic range of the pixels used in said weighted sum multiplied by a value equal to the sum of the respective weights used in said weighted sum.
Claim 22	
22	A method according to claim 21, wherein the dynamic range of said first intermediate value is defined by the number of bits used to represent the respective pixels used in said weighted sum plus the number of bits required to represent the sum of the respective weights used in said weighted sum.
Claim 23	
23	A method according to claim 21, comprising truncating the first intermediate value by mathematically dividing the first intermediate value by a first scale factor, said first scale factor having a value equal to the sum of the respective weights used in said weighted sum, thereby forming a sub-pixel value with a dynamic range equal to the predetermined dynamic range of said pixels.
Claim 24	
24	A method according to claim 23, comprising using a truncated first intermediate sub-pixel value when interpolating a value for a sub-pixel having co-ordinates with odd values of K and L.
Claim 25	
25	A method according to claim 21, comprising interpolating sub-pixel values for sub-pixels having co-ordinates with even values

	of K and L, using a weighted sum comprising first intermediate values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and corresponding first intermediate values for sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, to produce a second intermediate value, said second intermediate value having a dynamic range equal to the dynamic range of the first intermediate values used in said weighted sum, multiplied by a value equal to the sum of the respective weights used in said weighted sum.
Claim 26	
26	A method according to claim 25, comprising truncating the second intermediate value by mathematically dividing the second intermediate value by a second scale factor, said second scale factor having a value equal to the sum of the respective weights used in the weighted sum of claim 21 multiplied by the sum of the respective weights used in the weighted sum of claim 25, thereby forming a sub-pixel value with a dynamic range equal to the predetermined dynamic range of said pixels.
Claim 28	
28	A method according to claim 21, comprising interpolating sub-pixel values for sub-pixels having co-ordinates with even values of K and L, using a weighted sum comprising first intermediate values for sub-pixels having co-ordinates with K equal to zero and L equal to an even value and corresponding first intermediate values for sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions to produce a second intermediate value, said second intermediate value having a dynamic range equal to the dynamic range of the first intermediate values used in said weighted sum, multiplied by a value equal to the sum of the respective weights used in said weighted sum.

Claim 29	
29	A method according to claim 28, comprising truncating the second intermediate value by mathematically dividing the second intermediate value by a second scale factor, said second scale factor having a value equal to the sum of the respective weights used in the weighted sum of claim 21 multiplied by the sum of the respective weights used in the weighted sum of claim 28, thereby forming a sub-pixel value with a dynamic range equal to the dynamic range of said pixels.
Claim 31	
31[pre][a]	A method for quarter resolution sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
31[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/4$, $L/4$, K and L being positive integers having respective values between zero and 4, the method comprising causing an apparatus to:
31[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with K and L equal to 1 or 3, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $2/4$, $2/4$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with K and L equal to zero, 2 or 4, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $2/4$, $2/4$ and the nearest neighbouring pixel;
31[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to 2 and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to 2, used in the interpolation of the sub-pixels having co-ordinates with K and L

	equal to 1 or 3, using weighted sums of the values of pixels located in rows and columns respectively; and
31[c]	interpolate a sub-pixel value for the sub-pixel having co-ordinates with K and L equal to 2, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with K and L equal to 1 or 3, using a predetermined choice of either a weighted sum of the values of the sub-pixel having co-ordinates with K equal to 2 and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the value of the sub-pixel having co-ordinates with K equal to zero and L equal to 2 and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 32	
32[pre][a]	A method for eighth resolution sub-pixel value interpolation to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
32[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically with the rectangular bounded region using the co-ordinate notation $K/8$, $L/8$, K and L being positive integers having respective values between zero and 8, the method comprising causing an apparatus to:
32[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with K and L equal to 1, 3, 5 or 7, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $4/8$, $4/8$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with K and L equal to zero, 2, 4, 6 or 8, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $4/8$, $4/8$ and the nearest neighbouring pixel;

32[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to 2, 4 or 6 and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to 2, 4 or 6, used in the interpolation of the sub-pixels having co-ordinates with K and L equal to 1, 3, 5 or 7, using weighted sums of the values of pixels located in rows and columns respectively; and
32[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with K and L equal to 2, 4 or 6, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with K and L equal to 1, 3, 5 or 7, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to 2, 4 or 6 and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to 2, 4 or 6 and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 33	
33[pre][a]	An interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
33[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:
33[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-

	ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
33[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
33[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 36	
36	An interpolator according to claim 33, wherein the interpolator is configured to use a first and a second weight in the weighted average chosen to interpolate the sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, wherein when the chosen weighted average is that involving the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, the interpolator is configured to select the relative magnitudes of the first and second weights to be inversely proportional to the respective straight-line diagonal distances of the sub-pixels having co-ordinates with even values of K and L to

	the sub-pixel having co-ordinates with odd values of K and L whose value is being interpolated.
Claim 37	
37	An interpolator according to claim 36, wherein the interpolator is configured to use first and second weights with equal values when the sub-pixels having co-ordinates with even values of K and L are equidistant from the sub-pixel having co-ordinates with odd values of K and L whose value is being interpolated.
Claim 38	
38	An interpolator according to claim 33, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions when the value of a sub-pixel having co-ordinates with K equal to an even value and L equal to an odd value is also required.
Claim 39	
39	An interpolator according to claim 33, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions when the value of a sub-pixel having co-ordinates with K equal to an odd value and L equal to an even value is also required.
Claim 40	
40	An interpolator according to claim 33, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having

	co-ordinates with K equal to an odd value O and L equal to an even value E, not including zero or 2^N , by taking an average of the value of a first sub-pixel having co-ordinates with K equal to the even value O-1 and L equal to E and a second sub-pixel having co-ordinates with K equal to the even value O+1 and L equal to E.
Claim 41	
41	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 1 and L equal to zero, by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to zero, and a sub-pixel having co-ordinates with K equal to 2 and L equal to zero.
Claim 42	
42	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^N-1 and L equal to zero, by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to zero, and a sub-pixel having co-ordinates with K equal to 2^N-2 and L equal to zero.
Claim 43	
43	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 1 and L equal to 2^N , by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to 2 and L equal to 2^N .
Claim 44	
44	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^N-1 and L equal to 2^N , by taking an average of the value of a pixel having co-ordinates with K equal

	to 2^N and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to 2^{N-2} and L equal to 2^N .
Claim 45	
45	An interpolator according to claim 33, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value E, not including zero or 2^N , and L equal to an odd value O, by taking an average of the value of a first sub-pixel having co-ordinates with K equal to E and L equal to the even value O-1 and a second sub-pixel having co-ordinates with K equal to E and L equal to the even value O+1.
Claim 46	
46	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to zero and L equal to 1, by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to zero, and a sub-pixel having co-ordinates with K equal to zero and L equal to 2.
Claim 47	
47	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^N and L equal to 1, by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to zero, and a sub-pixel having co-ordinates with K equal to 2^N and L equal to 2.
Claim 48	
48	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to zero and L equal to 2^{N-1} , by taking an average of the value of a pixel having co-ordinates with K equal to zero and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to zero and L equal to 2^{N-2} .

Claim 49	
49	An interpolator according to claim 33, wherein the interpolator is configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with K equal to 2^N and L equal to 2^N-1 , by taking an average of the value of a pixel having co-ordinates with K equal to 2^N and L equal to 2^N , and a sub-pixel having co-ordinates with K equal to 2^N and L equal to 2^N-2 .
Claim 51	
51	An interpolator according to claim 33, wherein N is set equal to 2.
Claim 52	
52	An interpolator according to claim 33, wherein N is set equal to 3.
Claim 53	
53	An interpolator according to claim 33, wherein the pixels have a predetermined dynamic range, said predetermined dynamic range corresponding to the range of values said pixels can take.
Claim 54	
54	An interpolator according to claim 53, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and for sub-pixels having co-ordinates with K equal to zero and L equal to an even value, using weighted sums of the values of pixels located in rows and columns respectively, to produce a first intermediate value, said first intermediate value having a dynamic range equal to the predetermined dynamic range of the pixels used in said weighted sum multiplied by a value equal to the sum of the respective weights used in said weighted sum.
Claim 55	
55	An interpolator according to claim 54, wherein the dynamic range of said first intermediate value is defined by the number of bits used to represent the respective pixels used in said weighted sum

	plus the number of bits required to represent the sum of the respective weights used in said weighted sum.
Claim 56	
56	An interpolator according to claim 54, wherein the interpolator is configured to truncate the first intermediate value by mathematically dividing the first intermediate value by a first scale factor, said first scale factor having a value equal to the sum of the respective weights used in said weighted sum, thereby forming a sub-pixel value with a dynamic range equal to the predetermined dynamic range of said pixels.
Claim 57	
57	An interpolator according to claim 56, wherein the interpolator is configured to use a truncated first intermediate sub-pixel value when interpolating a value for a sub-pixel having co-ordinates with odd values of K and L.
Claim 58	
58	An interpolator according to claim 54, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, using a weighted sum comprising first intermediate values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and corresponding first intermediate values for sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, to produce a second intermediate value, said second intermediate value having a dynamic range equal to the dynamic range of the first intermediate values used in said weighted sum, multiplied by a value equal to the sum of the respective weights used in said weighted sum.
Claim 59	
59	An interpolator according to claim 58, wherein the interpolator is configured to truncate the second intermediate value by mathematically dividing the second intermediate value by a second scale factor, said second scale factor having a value equal

	to the sum of the respective weights used in the weighted sum of claim 54 multiplied by the sum of the respective weights used in the weighted sum of claim 58, thereby forming a sub-pixel value with a dynamic range equal to the predetermined dynamic range of said pixels.
Claim 61	
61	An interpolator according to claim 54, wherein the interpolator is configured to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, using a weighted sum comprising first intermediate values for sub-pixels having co-ordinates with K equal to zero and L equal to an even value and corresponding first intermediate values for sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions to produce a second intermediate value, said second intermediate value having a dynamic range equal to the dynamic range of the first intermediate values used in said weighted sum, multiplied by a value equal to the sum of the respective weights used in said weighted sum.
Claim 62	
62	An interpolator according to claim 61, wherein the interpolator is configured to truncate the second intermediate value by mathematically dividing the second intermediate value by a second scale factor, said second scale factor having a value equal to the sum of the respective weights used in the weighted sum of claim 54 multiplied by the sum of the respective weights used in the weighted sum of claim 61, thereby forming a sub-pixel value with a dynamic range equal to the predetermined dynamic range of said pixels.
Claim 64	
64[pre][a]	A video encoder comprising an interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between

	the corners, the pixels and sub-pixels being configured for display in rows and columns,
64[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:
64[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
64[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
64[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having

	corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 65	
65[pre][a]	A codec comprising a video encoder according to claim 64 and a video decoder comprising an interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
65[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:
65[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
65[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and

65[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 66	
66	A communications terminal comprising a video encoder according to claim 64.
Claim 67	
67	A communications terminal according to claim 66, wherein the communications terminal is a wireless terminal.
Claim 68	
68[pre][a]	A still image encoder comprising an interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
68[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:

68[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
68[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
68[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 69	
69[pre][a]	A codec comprising a still image encoder according to claim 68 and a still image decoder comprising an interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between

	the corners, the pixels and sub-pixels being configured for display in rows and columns,
69[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:
69[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
69[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
69[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having

	corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 70	
70	A communications terminal comprising a still image encoder according to claim 68.
Claim 71	
71[pre][a]	A video decoder comprising an interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
71[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:
71[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
71[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-

	ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
71[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 72	
72	A communications terminal comprising a video decoder according to claim 71.
Claim 75	
75[pre][a]	An interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
75[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator comprising:
75[a]	circuitry configured to interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the

	value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
75[b]	circuitry configured to interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
75[c]	circuitry configured to interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 76	
76[pre][a]	An interpolator for quarter resolution sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
76[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded

	region using the co-ordinate notation $K/4$, $L/4$, K and L being positive integers having respective values between zero and 4, the interpolator being configured to:
76[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with K and L equal to 1 or 3, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $2/4$, $2/4$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with K and L equal to zero, 2 or 4, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $2/4$, $2/4$ and the nearest neighbouring pixel;
76[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to 2 and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to 2, used in the interpolation of the sub-pixels having co-ordinates with K and L equal to 1 or 3, using weighted sums of the values of pixels located in rows and columns respectively; and
76[c]	interpolate a sub-pixel value for the sub-pixel having co-ordinates with K and L equal to 2, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with K and L equal to 1 or 3, using a predetermined choice of either a weighted sum of the values of the sub-pixel having co-ordinates with K equal to 2 and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the value of the sub-pixel having co-ordinates with K equal to zero and L equal to 2 and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 77	
77	A video encoder or still image encoder comprising an interpolator according to claim 76.

Claim 78	
78	A communications terminal comprising a video encoder or still image encoder according to claim 77.
Claim 79	
79 <pre>[a]</pre>	An interpolator for eighth resolution sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
79 <pre>[b]</pre>	pixel and sub-pixel locations in the rows and columns being representable mathematically with the rectangular bounded region using the co-ordinate notation $K/8$, $L/8$, K and L being positive integers having respective values between zero and 8, the interpolator being configured to:
79[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with K and L equal to 1, 3, 5 or 7, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $4/8$, $4/8$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with K and L equal to zero, 2, 4, 6 or 8, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $4/8$, $4/8$ and the nearest neighbouring pixel;
79[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to 2, 4 or 6 and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to 2, 4 or 6, used in the interpolation of the sub-pixels having co-ordinates with K and L equal to 1, 3, 5 or 7, using weighted sums of the values of pixels located in rows and columns respectively; and
79[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with K and L equal to 2, 4 or 6, used in the interpolation of sub-

	pixel values for the sub-pixels having co-ordinates with K and L equal to 1, 3, 5 or 7, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to 2, 4 or 6 and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to 2, 4 or 6 and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 80	
80	A video decoder or still image decoder comprising an interpolator according to claim 79.
Claim 81	
81	A communications terminal comprising a video decoder or still image decoder according to claim 80.
Claim 82	
82[pre][a]	A computer program for sub-pixel value interpolation, embodied on a computer readable storage medium, the computer program being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
82[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the computer program comprising:
82[a]	computer program code for interpolating a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average

	of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
82[b]	computer program code for interpolating sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
82[c]	computer program code for interpolating sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.
Claim 83	
83[pre][a]	A network apparatus comprising an interpolator for sub-pixel value interpolation, the interpolator being configured to determine values for sub-pixels situated within a rectangular bounded region defined by four corner pixels with no intermediate pixels between the corners, the pixels and sub-pixels being configured for display in rows and columns,
83[pre][b]	pixel and sub-pixel locations in the rows and columns being representable mathematically within the rectangular bounded

	region using the co-ordinate notation $K/2^N$, $L/2^N$, K and L being positive integers having respective values between zero and 2^N , N being a positive integer greater than one and representing a particular degree of sub-pixel value interpolation, the interpolator being configured to:
83[a]	interpolate a sub-pixel value for a sub-pixel having co-ordinates with odd values of K and L, according to a predetermined choice of either a weighted average of the value of a nearest-neighbouring pixel and the value of the sub-pixel situated at co-ordinates $\frac{1}{2}$, $\frac{1}{2}$, or a weighted average of the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}$, $\frac{1}{2}$ and the nearest neighbouring pixel;
83[b]	interpolate sub-pixel values for sub-pixels having co-ordinates with K equal to an even value and L equal to zero and sub-pixels having co-ordinates with K equal to zero and L equal to an even value, used in the interpolation of the sub-pixels having co-ordinates with odd values of K and L, using weighted sums of the values of pixels located in rows and columns respectively; and
83[c]	interpolate sub-pixel values for sub-pixels having co-ordinates with even values of K and L, used in the interpolation of sub-pixel values for the sub-pixels having co-ordinates with odd values of K and L, using a predetermined choice of either a weighted sum of the values of sub-pixels having co-ordinates with K equal to an even value and L equal to zero and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions, or a weighted sum of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value and the values of sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.

I. INTRODUCTION

Amazon.com, Inc. and Amazon.com Services LLC (“Amazon” or “Petitioner”) request *inter partes* review (“IPR”) of claims 1, 4-17, 19-26, 28-29, 31-33, 36-49, 51-59, 61-62, and 64-83 (“Challenged Claims”) of U.S. Patent No. 8,036,273 (“’273 patent”; EX-1001).

II. GROUNDS FOR STANDING

Petitioner certifies that the ’273 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 on Amazon of Patent Owner’s district court complaint alleging infringement of the ’273 patent. EX-1020.

III. IDENTIFICATION OF CHALLENGE

A. Prior Art

Prior Art
TML6 (H.26L Test Model Long Term Number 6) , publicly available by March 28, 2001, prior art at least under §102(a) ¹ (EX-1005)
Fandrianto (U.S. Patent No. 5,594,813) , issued on January 14, 1997, prior art at least under §102(b) (EX-1006)

B. Grounds

Petitioner respectfully requests review and cancellation under 35 U.S.C. §311 of the Challenged Claims in view of:

¹ All references to 35 U.S.C. §§102, 103 are to the pre-AIA statutory framework.

Ground	Claims	References
1 (§103)	1, 4-17, 19-26, 28-29, 31, 33, 36-49, 51, 53-59, 61-62, 64-78, 82-83	TML6, Fandrianto
2 (§103)	32, 52, 79-81	TML6, Fandrianto, knowledge of a POSITA

IV. '273 PATENT OVERVIEW

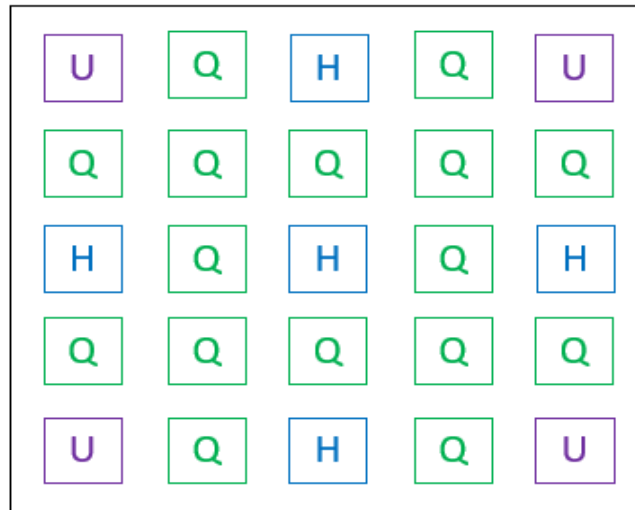
A. Technology Background

Video data consist of sequences of still images, or “frames,” comprising rows and columns of “pixels.” EX-1001, 1:43-45. “Motion compensated temporal prediction” is a video coding technique that exploits the redundancy typically present between successive frames in a video sequence. EX-1001, 2:62-3:2. Motion-predicted video frames are represented by “motion vectors” that indicate the displacement of pixels in a current frame relative to their position in another (“reference”) frame. EX-1001, 4:18-26; Ramchandran, ¶¶21-25. Motion vectors are determined by comparing pixels in the current frame with pixels within a “search region” of the reference frame. EX-1001, 4:54-60, Figure 4(c).

Objects—i.e., pixel groups/blocks—may move between frames by distances that are not exact pixel locations but are more closely represented by fractions of the distance between pixels. Ramchandran, ¶26. In such cases, the values of virtual “sub-pixels” must be “interpolated” using actual pixel values. Ramchandran, ¶27. The benefits of sub-pixel interpolation were well-known by the '273 patent's earliest

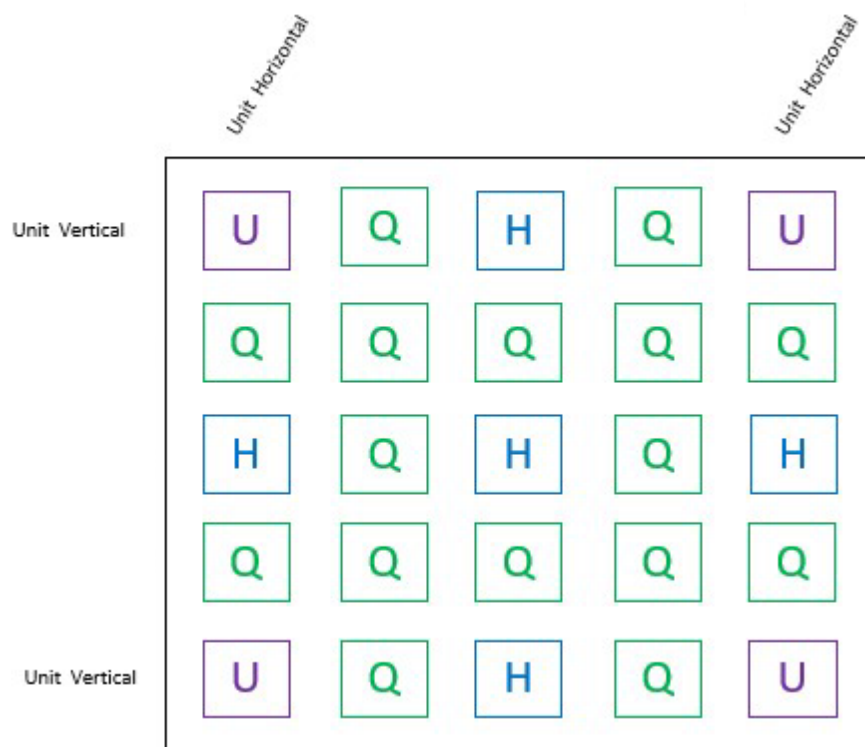
possible priority date. VCEG-L45, 15; Fandrianto, 5:26-28; Ramchandran, ¶¶29-33, 43.

For clarity, certain terminology is addressed below. The figure below² shows four actual pixels, referred to as **unit pixels U** (purple). **Half (1/2) sub-pixels H** (blue) are located halfway (horizontally, vertically, or both) between unit pixels. **Quarter (1/4) sub-pixels Q** (green) are located quarter-way between unit pixels.

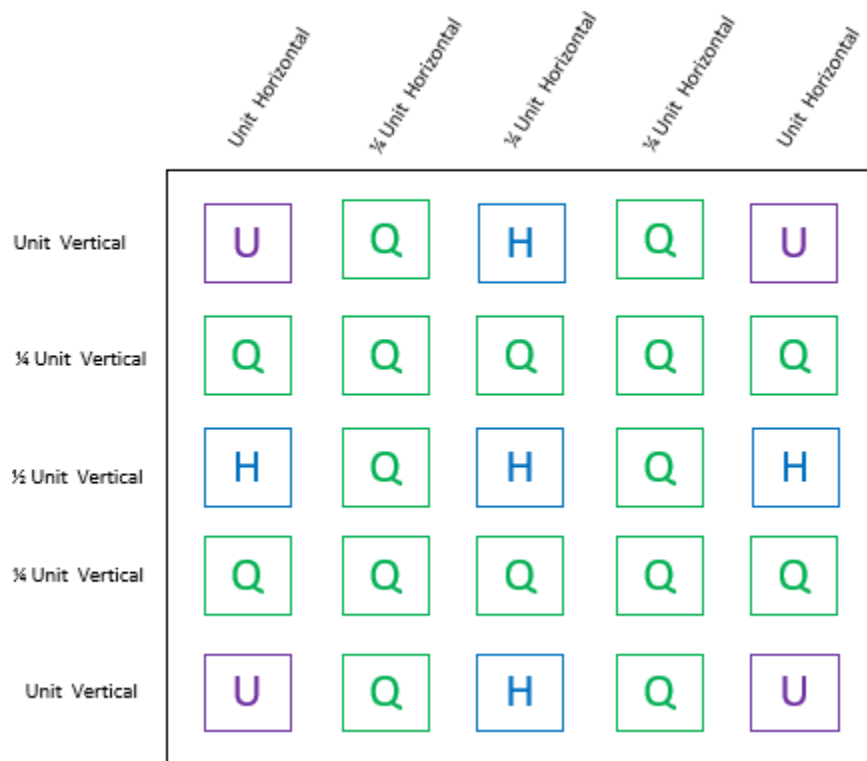


Ramchandran, ¶28. As shown below, **unit horizontal** location refers to a pixel/sub-pixel situated in a **column** that contains unit pixels (purple). Similarly, **unit vertical** location refers to a pixel/sub-pixel situated in a **row** that contains unit pixels. Thus, a unit pixel is situated at both unit horizontal and unit vertical locations.

² Figures like the one below that include a citation to Dr. Ramchandran's declaration (EX-1003) have been created to graphically illustrate certain concepts.

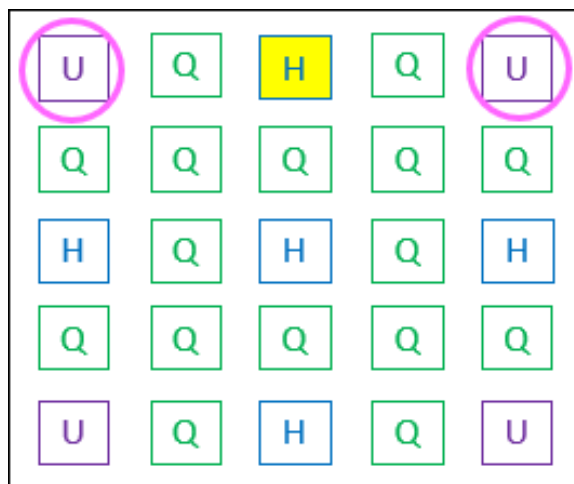


Ramchandran, ¶¶28-29. **Half (1/2) unit horizontal location** refers to a position halfway between two unit horizontal locations and **half (1/2) unit vertical location** refers to a position halfway between two unit vertical locations. **Quarter (1/4) unit horizontal location** refers to a position quarter-way between two unit horizontal locations and **quarter (1/4) unit vertical location** refers to a position quarter-way between two unit vertical locations.



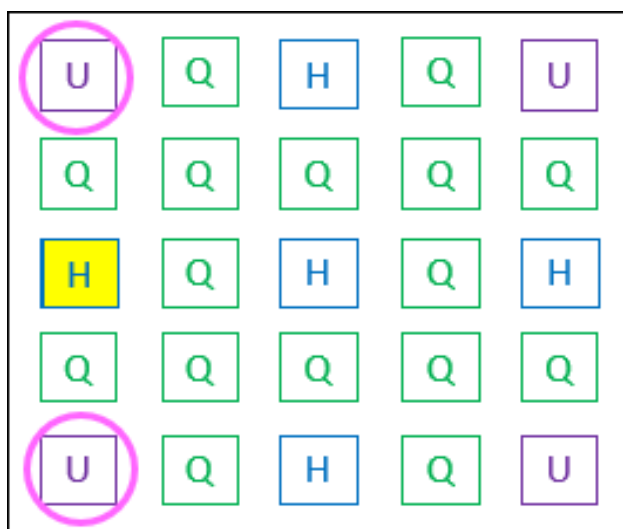
Ramchandran, ¶¶28-29.

Sub-pixel values can be interpolated using pixel/sub-pixel values along a horizontal, vertical, or diagonal line. As shown below, the highlighted half sub-pixel H (blue)—situated at the half unit horizontal, unit vertical location—can be **horizontally interpolated** using the two unit pixels (purple; circled) in the **same row**.



$\frac{1}{2}$ unit horizontal
unit vertical

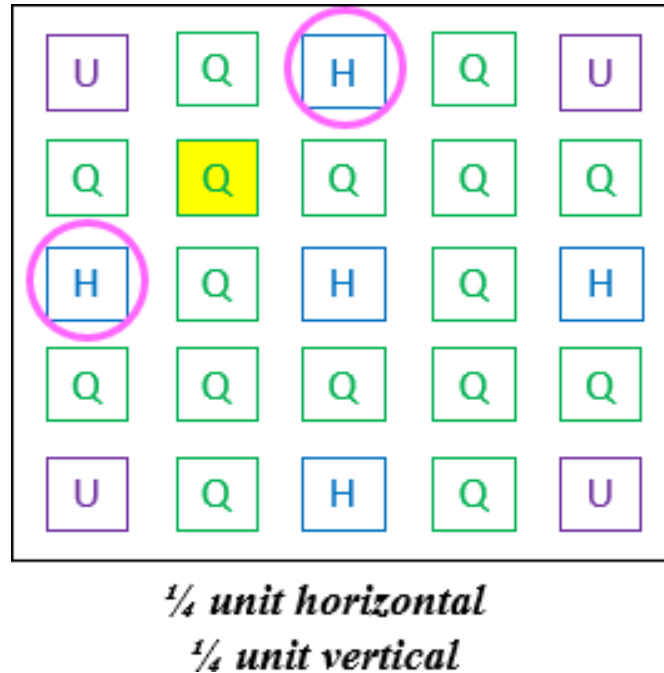
Ramchandran, ¶30. Likewise, as shown below, the highlighted half sub-pixel H (blue)—situated at the unit horizontal, half unit vertical location—can be **vertically interpolated** using the two unit pixels (purple; circled) in the **same column**.



unit horizontal
 $\frac{1}{2}$ unit vertical

Ramchandran, ¶31. Finally, as shown below, the highlighted quarter sub-pixel Q (green)—situated at the quarter unit horizontal, quarter unit vertical (or “quarter-

quarter”)³ location—can be **diagonally interpolated** using previously interpolated half sub-pixels H (blue; circled).



Ramchandran, ¶32.

B. The Alleged Problem in the Prior Art

The '273 patent admits that interpolating half and quarter sub-pixels was known. EX-1001, 6:35-41, 7:1-19, 8:38-9:3, 9:65-12:64; Ramchandran, ¶34. The Background describes TML6's sub-pixel interpolation method in conjunction with

³ This Petition uses the shorthand “quarter-quarter sub-pixels” to refer to sub-pixels situated at the quarter unit horizontal and quarter unit vertical locations, and similarly uses “half-half sub-pixels” to refer to sub-pixels situated at the half unit horizontal and half unit vertical locations.

Figure 13(a), below, and explains that quarter sub-pixels “d” and “e” (green) are calculated using horizontal or vertical interpolation.

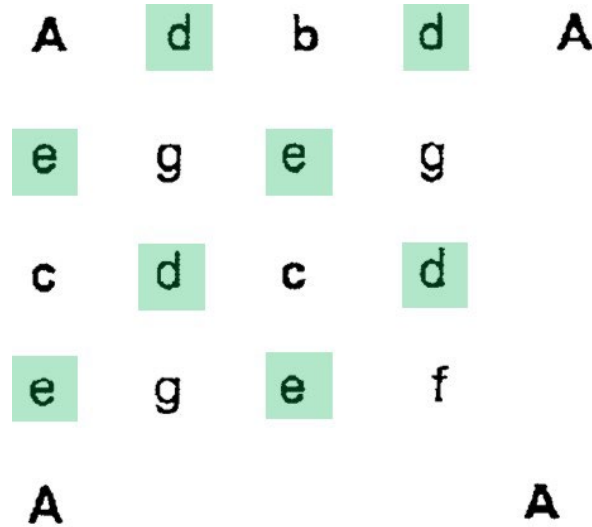


Fig. 13(a)

EX-1001, Figure 13(a),⁴ 11:64-12:33, 28:36-37.

According to the '273 patent, TML6 does not calculate the highlighted quarter sub-pixel “g” (green) (below) using horizontal or vertical interpolation. Instead, sub-pixel “g” is interpolated based on the following four pixels/sub-pixels: pixel “A,” sub-pixel “b,” and sub-pixels “c” (circled).

⁴ Color annotations have been added.

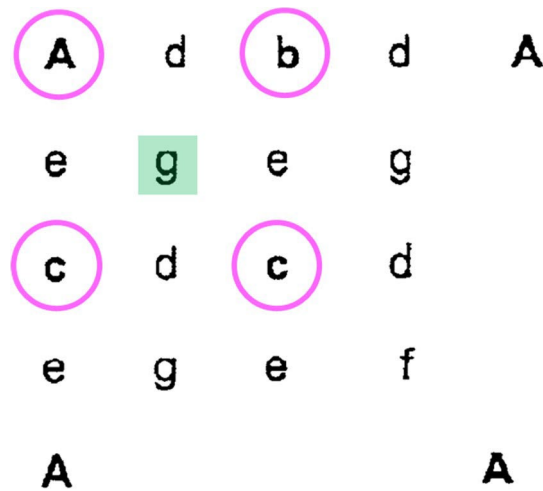


Fig. 13(a)

EX-1001, Figure 13(a), 8:38-48, 12:34-64; Ramchandran, ¶¶35-38. The '273 patent asserts, however, interpolating certain sub-pixels based on four other pixels/sub-pixels is too computationally intensive. *See* EX-1001, 44:9-34.

C. The Alleged Invention of the '273 Patent

To address this purported problem, as illustrated in Figure 14(a) below, the '273 patent claims as its allegedly novel concept using two diagonally opposite half sub-pixels “b” (blue) to interpolate quarter sub-pixel “h” (green).

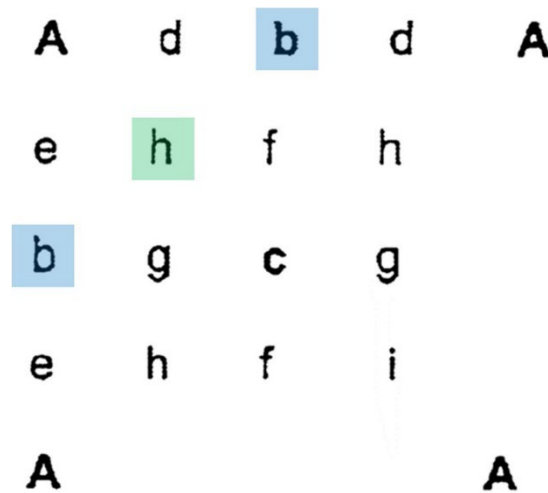


Fig. 14(a)

EX-1001, Figure 14(a), 28:42-43, 42:53-43:7; Ramchandran, ¶¶39-40. This technique allegedly improves on TML6 because it “does not require high precision arithmetic to be used in the calculation of all sub-pixels.” EX-1001, 44:9-34; Ramchandran, ¶40.

V. LEVEL OF ORDINARY SKILL

A person of ordinary skill in the art (“POSITA”) at the time of the alleged invention of the ’273 patent would have had a (1) Bachelor’s degree in electrical engineering, computer engineering, computer science, or a comparable field of study such as physics, and (2) approximately two to three years of practical experience

with video encoding/decoding. Additional experience can substitute for the level of education, and vice-versa.⁵ Ramchandran, ¶41.

VI. CLAIM CONSTRUCTION

Claims should be construed “in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.” 37 C.F.R. §42.100(b); *see also Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). Petitioner is unaware of any “prior claim construction determination” related to the ’273 patent. 37 C.F.R. §42.100(b).

Petitioner has not proposed any terms for construction because no constructions are necessary to resolve any disputes identified in this Petition.⁶ *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999).

⁵ Although the definition of a POSITA proposed in the four prior IPRs challenging the ’273 patent (IPR2021-00040, IPR2021-00041, IPR2021-00042, and IPR2021-00043) differs slightly from the definition proposed here, the Challenged Claims are invalid under both definitions. Ramchandran, ¶42.

⁶ If trial is instituted, Petitioner reserves the right to address construction of any terms raised at trial, or in the related district court action.

VII. PRIOR ART OVERVIEW

A. TML6

TML6 (EX-1005) excerpts the “Test Model Long Term Number 6” source code authored by the Video Coding Experts Group (“VCEG”) of the International Telecommunication Union Telecommunication Standardization Section (“ITU-T”). EX-1001, 8:43-53; Wenger, ¶¶15, 21; Ramchandran, ¶44. Dr. Stephan Wenger, a VCEG member who authored portions of TML6, explains TML6 comprises a subset of files that together make up the fully functional source code of a working video encoder and decoder. Wenger, ¶¶19-21. These files were packaged in a single compressed file named “*tml6pre.zip*” and made publicly available via VCEG’s FTP server by March 28, 2001. Wenger, ¶¶18-38.

As Dr. Ramchandran explains, except for “readme.txt,” the files comprising EX-1005 originate from subfolders in “*tml6pre.zip*” named “*lencod*,” “*ldecod*,” and “*bin*” containing files with executable code. Ramchandran, ¶46. Each page of EX-1005 indicates the file name and location within the file structure of each file at the top left corner. They are listed below with corresponding page ranges:

File	TML6 (EX-1005) Pages
/tml6pre.zip/readme.txt	1-2
/tml6pre.zip/lencod/inc/image.h	3
/tml6pre.zip/lencod/src/block.c	4-34
/tml6pre.zip/lencod/inc/global.h	35-47

File	TML6 (EX-1005) Pages
/tml6pre.zip/lencod/src/ image.c	48-68
/tml6pre.zip/lencod/src/ lencod.c	69-100
/tml6pre.zip/lencod/src/ macroblock.c	101-128
/tml6pre.zip/ldecod/inc/ global.h	129-134
/tml6pre.zip/ldecod/inc/ image.h	135-136
/tml6pre.zip/ldecod/src/ image.c	137-154
/tml6pre.zip/ldecod/src/ ldecod.c	155-163
/tml6pre.zip/ldecod/src/ macroblock.c	164-183

TML6 excerpts VCEG’s bundled, publicly-available source code. Wenger, ¶¶15, 18-38; Ramchandran, ¶¶45-46. Treating TML6 as a single prior art reference is thus appropriate. *Avi Networks, Inc. v. Citrix Sys., Inc.*, IPR2019-00844, Paper 19, 13 (Oct. 1, 2019) (finding “four source code files...[which] were ‘distributed together in a single compressed file’...constitute a single prior art printed publication under [§102(b)]”); see *MacSports, Inc. v. Idea Nuevo, Inc.*, IPR2018-01006, Paper 6, 14 (Nov. 13, 2018).

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). TML6 was made available on a public VCEG FTP server by March 28, 2001. Wenger, ¶¶18-38. Subscribers to the VCEG email reflector, including participants, students, and the public, were notified when files

were uploaded to the public FTP folder, from which files were frequently downloaded. Wenger, ¶¶20-22, 26-28. People tracking the latest video coding standard visited VCEG sites to ensure their products and services aligned with current standards. Wenger, ¶¶37-38. A POSITA would have known VCEG test models like TML6 included technologies potentially entering the developing standard. Wenger, ¶37. Dr. Wenger personally shared VCEG work with colleagues and students during the relevant time. Wenger, ¶35. TML6 is thus §102(a) prior art to the '273 patent.

TML6 implements methods of interpolating half and quarter sub-pixels. TML6, 57-59 (lines 538-632); Ramchandran, ¶¶47-108. As the '273 patent explains, TML6's sub-pixel interpolation scheme achieves "direct interpolation...by using intermediate values which are neither rounded nor clipped." EX-1001, 11:25-31.

The '273 patent describes the "interpolation method according to TML6 as applied in the encoder...with reference to FIGS. 13[a-c]." EX-1001, 11:31-13:27.

First, "[t]he value for...½ resolution sub-pixel b [blue] in FIG. 13a, is obtained by first calculating an intermediate value b using a six tap filter...based upon the values of the 6 pixels (A₁ to A₆) [purple] situated in a row...symmetrically about b, as shown in FIG. 13b."

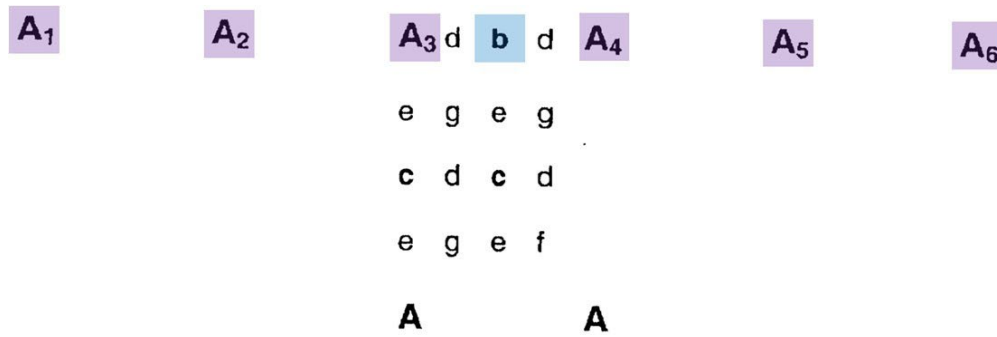
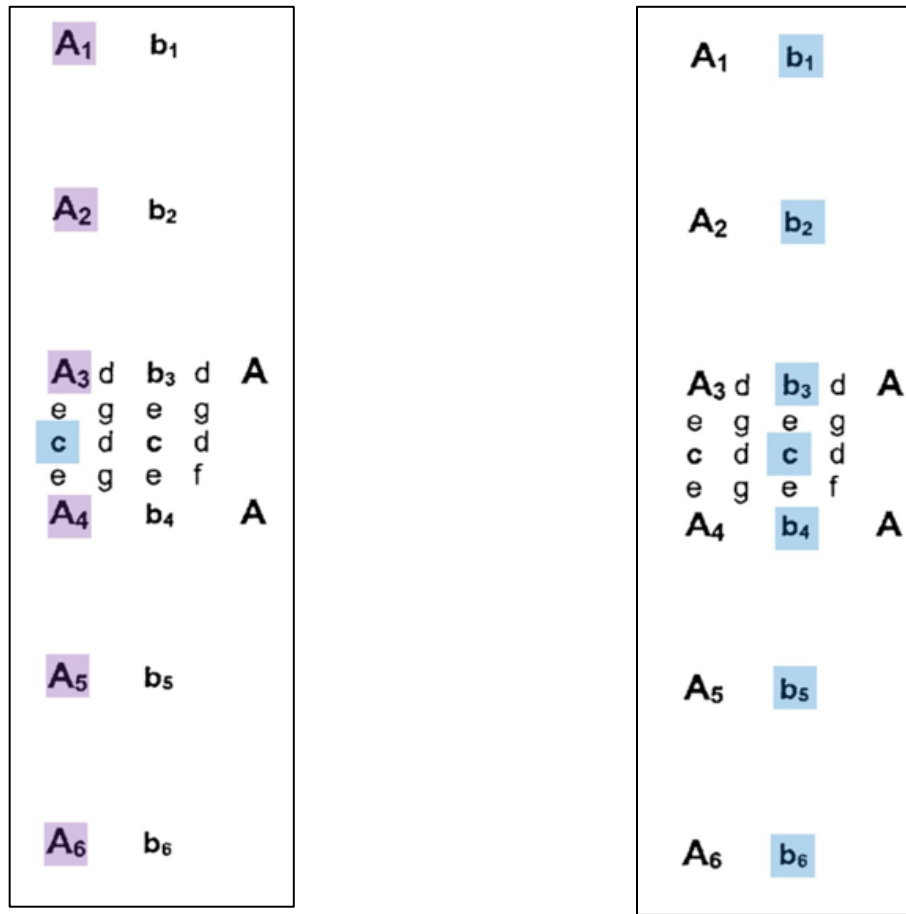


Figure13(b)

EX-1001, Figure 13(b), 11:35-45. As described for 1[a], TML6 confirms this interpolation method. TML6, 57-58 (lines 563-575, 588); Ramchandran, ¶¶48, 65-69.

Second, referring to FIG. 13(c) (below), the '273 patent explains “[v]alues for the $\frac{1}{2}$ resolution sub-pixels labelled c [blue] are obtained by first calculating intermediate values c ...for the $\frac{1}{2}$ resolution sub-pixel c ...[which] is calculated based upon the values of the 6 pixels (A_1 to A_6) [purple on the left] situated in a column...symmetrically about cSimilarly, an intermediate value c for the $\frac{1}{2}$ resolution sub-pixel c ...is calculated according to [values of the 6 previously-interpolated sub-pixels b_1 - b_6 (blue on the right)].”



EX-1001, Figure 13(c), 11:46-63. As described for 1[a]-1[b], TML6 confirms this interpolation method. TML6, 57-58 (lines 576-592); Ramchandran, ¶¶49, 70-79.

Third, referring to Figure 13(a) (below), “ $\frac{1}{4}$ resolution sub-pixels labelled g [green] are computed using the value of the nearest original pixel A [purple] and the intermediate values of the three nearest neighbouring $\frac{1}{2}$ resolution sub-pixels [blue].”

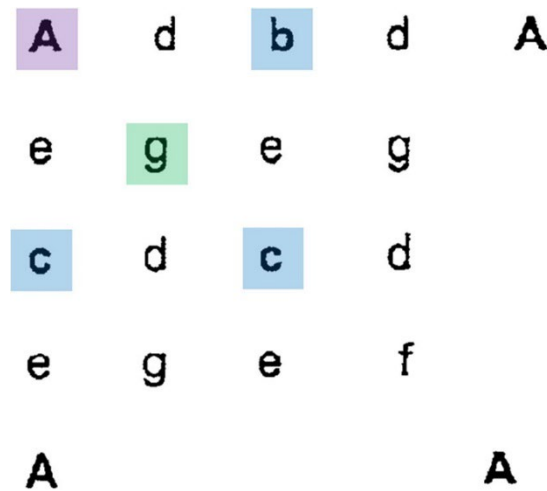


Fig. 13(a)

EX-1001, Figure 13(a), 12:34-40. As described for 1[c], TML6 confirms this interpolation method. TML6, 58 (lines 593-625); Ramchandran, ¶¶50, 96-101.

B. Fandrianto

Fandrianto is directed to a method and apparatus for “rapid and efficient motion vector searching” in a digital video compression system. Fandrianto, Abstract, 1:7-10, 1:43-67, 3:55-62; Ramchandran, ¶¶109-112. Fandrianto teaches “diagonal interpolation” to perform “motion vector searching with...quarter pixel accuracy.” *Id.*, 5:26-32, 7:53-55, Figures 3B, 4-5; Ramchandran, ¶¶113-117.

With reference to Figure 5 (below left), Fandrianto describes deriving highlighted quarter pixels Q1-Q8 in sequential order using the two-memory apparatus of Figure 3B (below right) comprising DPCM memory 130 and DP memory 124. DPCM memory 130 is a “search memory” in which “[t]he search

window...[for] a previously processed frame” is stored. DP memory 124 “store[s]...temporarily needed blocks,” e.g., “an input or image block” in P block 80 (pink) for “successively compar[ing]...with the search blocks in the search window” in DPCM memory 130.

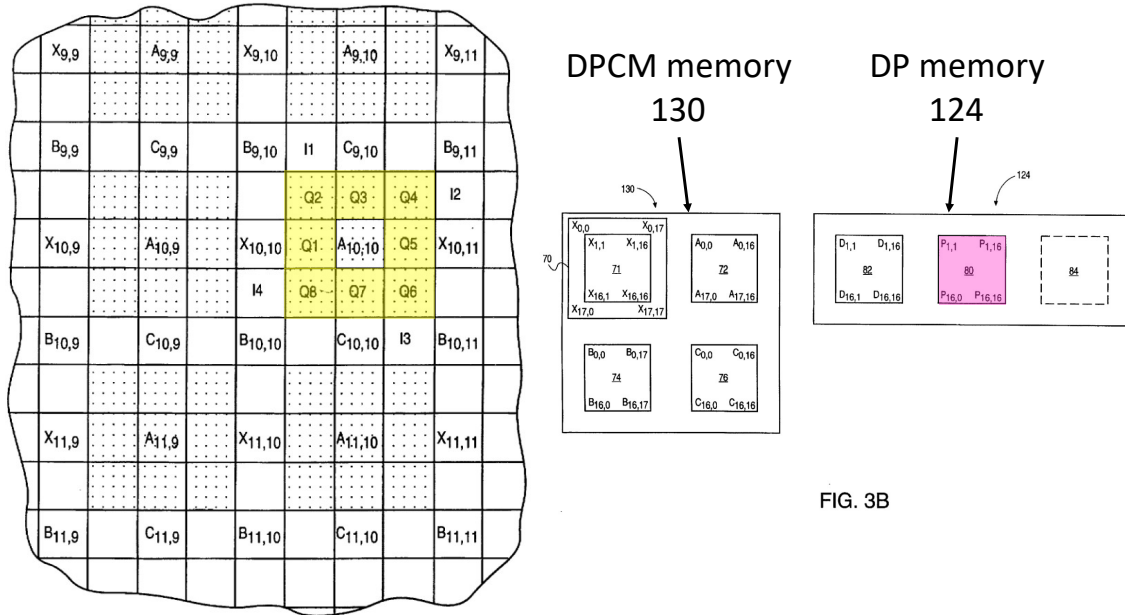


FIG. 3B

Fandrianto, Figure 5, Figure 3B, Abstract, 3:16-43, 3:57-66, 4:18-5:5, 5:26-32, 8:16-10:58, 12:9-15. Though Fandrianto focuses for simplicity on quarter pixels Q1-Q8 around sub-pixel A_{10,10}, its technique is block-based and calculates all the “highlighted areas about the elements A” in Figure 5, as described below. *See id.*, 8:24-28; Ramchandran, ¶156.

First, pixel block $X_{0,0}$ - $X_{17,17}$ (purple)⁷ is stored in “X” block 70 (purple) of DPCM memory 130.

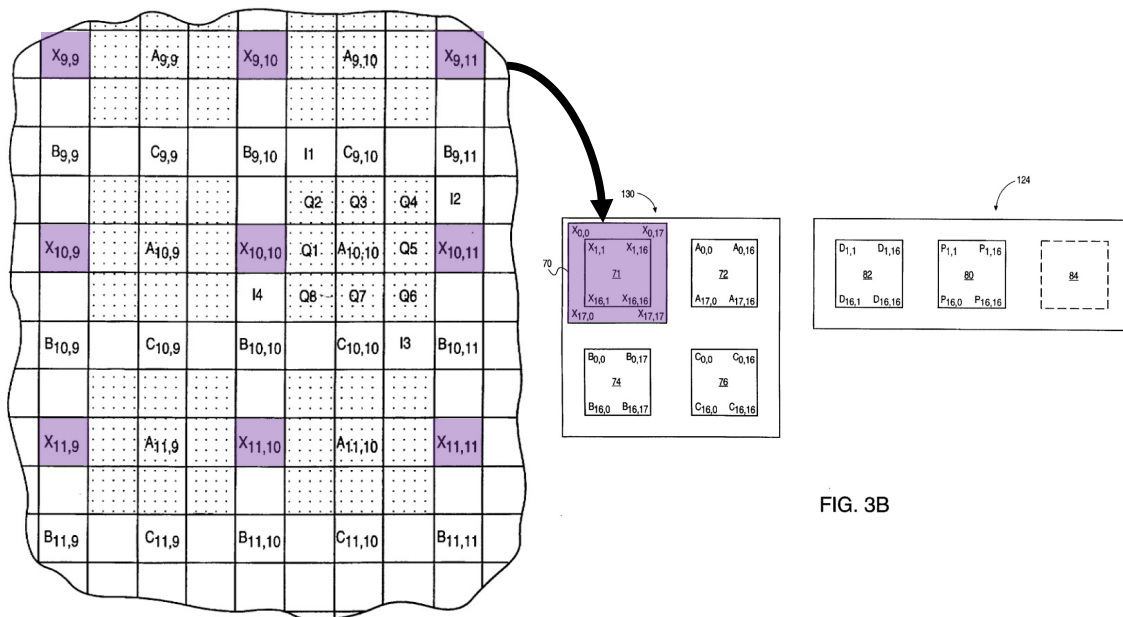


FIG. 3B

Fandrianto, Figure 5, Figure 3B, Abstract, 5:37-41. Next, “‘A’ block 72 [red], ‘B’ block 74 [blue], and ‘C’ block 76 [green] are interpolated from...X block 70[.]” The interpolated sub-pixels are annotated using like colors below.

⁷ In Fandrianto, the subscript notation “ $X_{r,c}$ ” indicates a range of pixel positions within a two-dimensional array “X” where the first subscript “r” denotes the row and the second subscript “c” denotes the column. Fandrianto, 5:51-55.

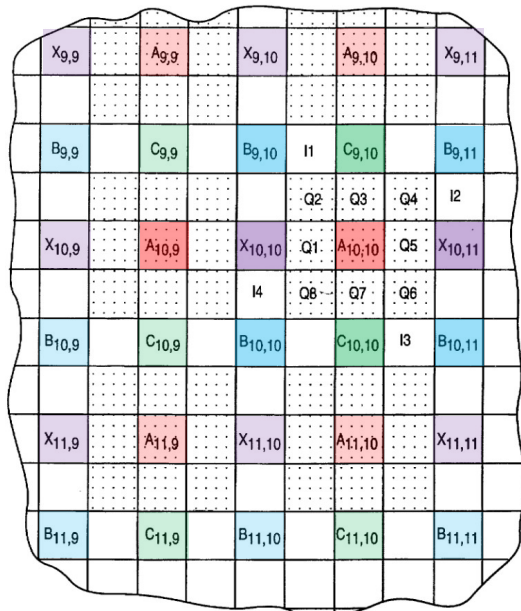


FIG. 5

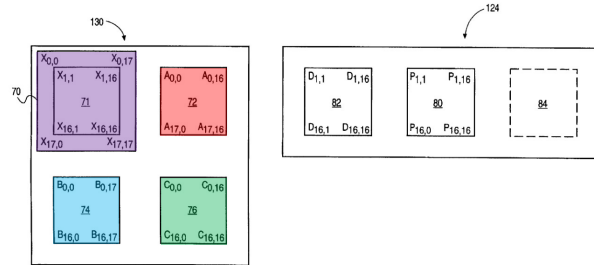


FIG. 3B

Fandrianto, Figure 5, Figure 3B, 5:41-44, 6:18-7:2. Fandrianto then describes an iterative process for sequentially generating quarter pixels Q1-Q8 wherein each quarter pixel is calculated based on the preceding quarter pixel and is stored in A block 72, progressively overwriting the preceding quarter pixel data. *Id.*, 8:29-10:50; Ramchandran, ¶157.

Fandrianto notes “that a great many permutations of the order of derivation of pixels Q1-Q8 may be practiced[.]” Fandrianto, 10:51-53. Accordingly, “the order in which [Q1-Q8] are generated and compared, and the selection of obsolete full and half pixel estimation search blocks to be overwritten are...discretionary” such that “[t]he order shown in FIG. 5” is only illustrative. *Id.*, 8:15-21; Ramchandran, ¶158.

As illustrated below, Fandrianto teaches “the Q2 pixels [yellow] may be derived from a **diagonal interpolation** of the data in the A block 72 [red] and the B

block 74 [blue].” Because Fandrianto previously established both the order in which Q1-Q8 are generated and compared and the selection of obsolete full and half pixel estimation search blocks to be overwritten are discretionary, A block 72 and B block 74 contain values for A pixels [red] and B pixels [blue], respectively. Ramchandran, ¶159.

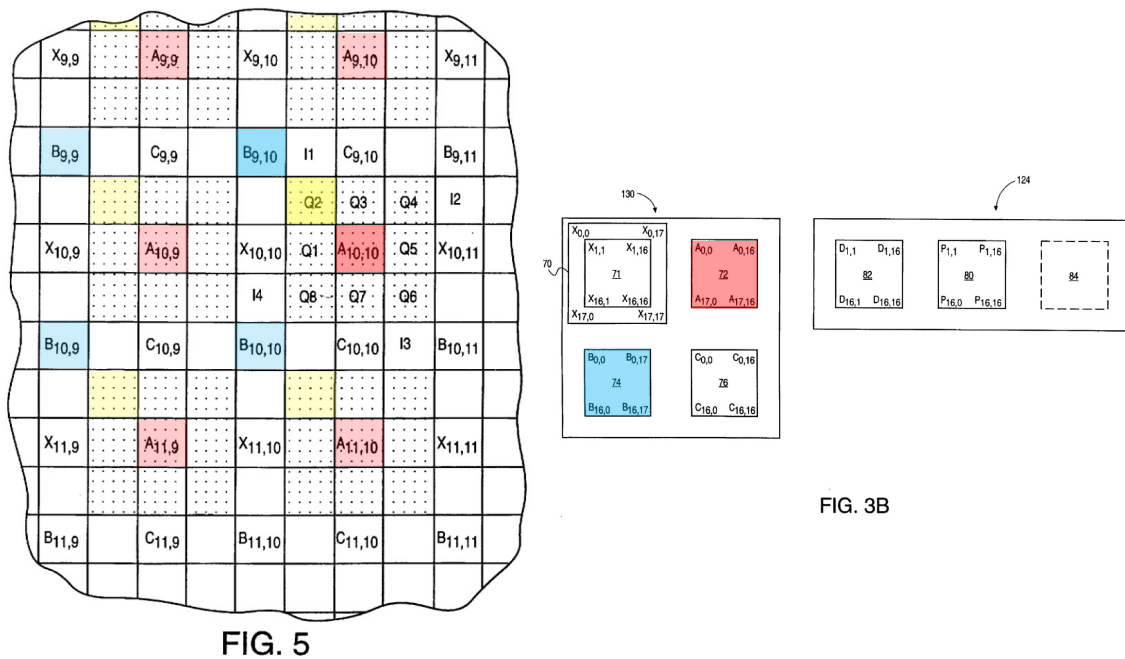


FIG. 3B

Fandrianto, Figures 5, 3B, 10:51-58.

VIII. DETAILED EXPLANATION OF GROUNDS

A. Ground 1: Obviousness over TML6 and Fandrianto (Claims 1, 4-17, 19-26, 28-29, 31, 33, 36-49, 51, 53-59, 61-62, 64-78, and 82-83)

1. Motivation to Combine

A POSITA would have been motivated to incorporate the teachings of Fandrianto (interpolating quarter-quarter sub-pixels) into TML6 (video coder with half and quarter sub-pixel interpolation) by implementing Fandrianto’s sub-pixel

interpolation method in TML6. Ramchandran, ¶¶119-120. TML6 and Fandrianto are in the same field (motion prediction video coding), address the same problem (objects moving between frames by sub-pixel distances), and solve that problem by interpolating half and quarter sub-pixels. Fandrianto, 5:26-28; TML6, 16-24 (lines 734-1180 (“*motion_search*” function)), 115-116 (lines 769-842 (“Motion vector search”)), 57-58 (lines 538-632 (sub-pixel interpolation)); EX-1001, 8:43-9:9, 11:33-13:29; Figures 13(a)-(c).

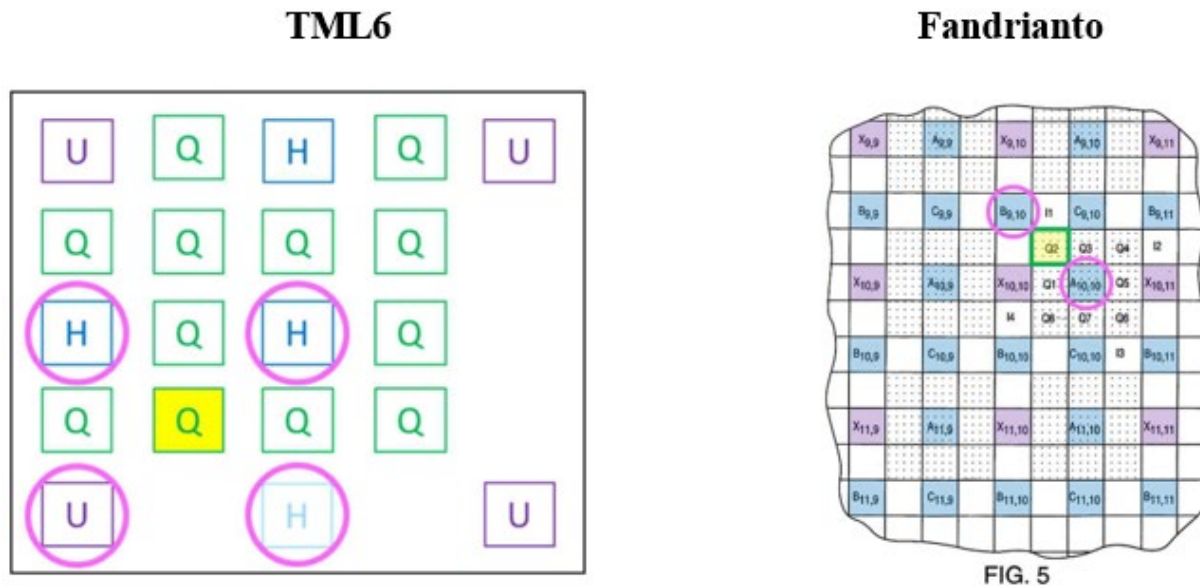
Fandrianto provides an express teaching that would have led a POSITA to combine its method of interpolating quarter-quarter sub-pixels with TML6. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 418-419 (2007). Fandrianto states: “Various interpolation techniques may be used in the derivation of a given quarter pixel estimation block, depending on various factors such as the amount of memory available[.]” Fandrianto, 7:32-35; *see id.*, 2:4-7, 3:55-4:6, 5:26-32, Figure 2. Fandrianto thus teaches a quarter-pixel interpolation technique suitable for a “limited memory” architecture. *Id.*, 7:36-37, 8:16-21, Figure 2. A POSITA would have implemented Fandrianto’s “[r]apid and efficient motion vector searching” technique in TML6 to realize Fandrianto’s stated benefit of “improved prediction accuracy” “while avoiding the need for additional memory.” *Id.*, 3:57-62, 5:1-5, 5:26-28.

Fandrianto discloses a “limited memory embodiment” capable of interpolating quarter sub-pixels using averages of values for pixels (“X-block”)

and/or already interpolated half sub-pixels (“A-block,” “B-block,” and “C-block”). *Id.*, Figure 5, 5:26-32, 7:27-10:58. A POSITA would have replaced the quarter sub-pixel interpolation in TML6’s *oneforthpix()* function (TML6, 58 (lines 593-625)) with Fandrianto’s technique that utilizes dedicated, optimized hardware rather than relying on a CPU to execute the TML6 code. Fandrianto, Figures 2, 5, 2:4-7, 3:55-4:6, 5:26-32, 7:27-10:58; Ramchandran, ¶132.

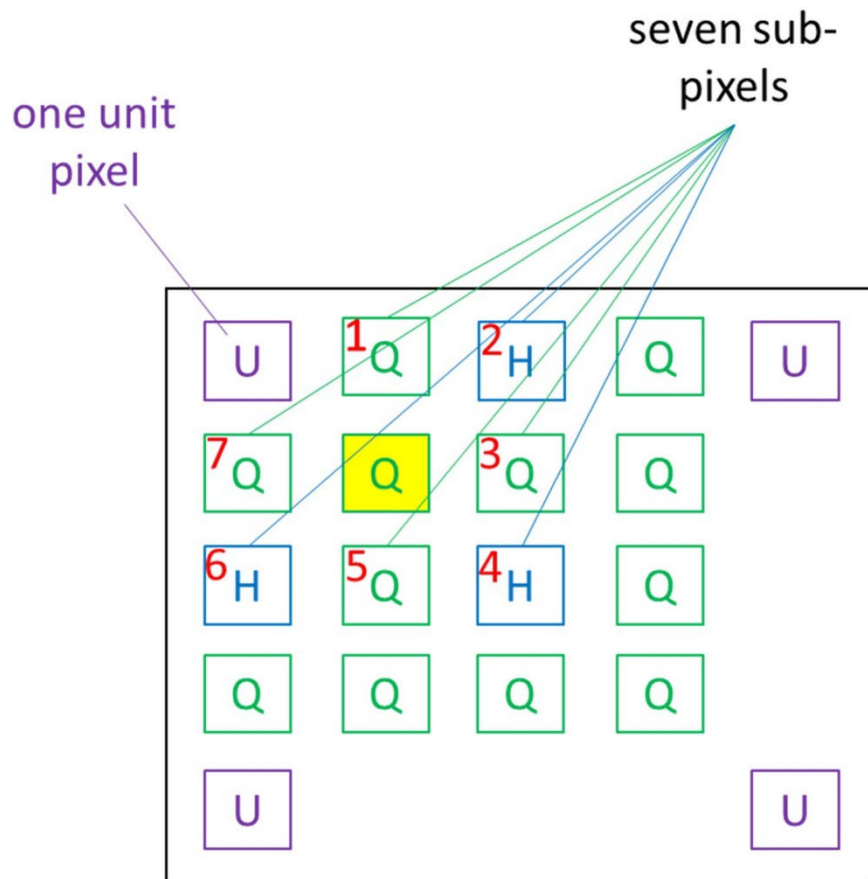
A POSITA would have recognized that Fandrianto’s method of interpolating quarter-quarter sub-pixels reflects a design choice between finite options with well-understood tradeoffs of improved accuracy and computational efficiency. *E.g.*, *Uber Techs., Inc. v. X One, Inc.*, 957 F.3d 1334, 1341 (Fed. Cir. 2020). Although TML6’s interpolation method relying on four values is arguably more accurate because it considers more pixels/sub-pixels near the quarter-quarter sub-pixel being interpolated, it is more computationally complex. TML6, 58 (lines 593-625). Conversely, Fandrianto’s interpolation method is less computationally intensive because it uses two values rather than four. Fandrianto, Figure 5, 10:51-58. Fandrianto discusses optimizing interpolation methods based on accuracy and efficiency considerations such as available memory, number of ports, and data buses. *Id.*, 7:32-37. A POSITA would have thus been motivated to use Fandrianto’s teachings as a design choice to increase computational efficiency. Ramchandran, ¶¶127-128.

The TML6-Fandrianto combination would have yielded predictable results because the combination is a simple substitution of Fandrianto's quarter-quarter sub-pixel interpolation method with TML6's quarter-quarter sub-pixel interpolation method to obtain a predictable result. As illustrated below, TML6 interpolates highlighted quarter-quarter sub-pixel based on the closest unit pixel (purple) and the three closest half sub-pixels (blue). TML6, 58 (lines 593-625). Fandrianto interpolates the highlighted quarter-quarter sub-pixel using the two closest diagonally-opposed half sub-pixels (blue). Fandrianto, Figure 5, 10:51-58.



A POSITA would have substituted Fandrianto's known interpolation method in TML6 to produce the predictable result of interpolating quarter-quarter sub-pixels. Ramchandran, ¶¶121-123.

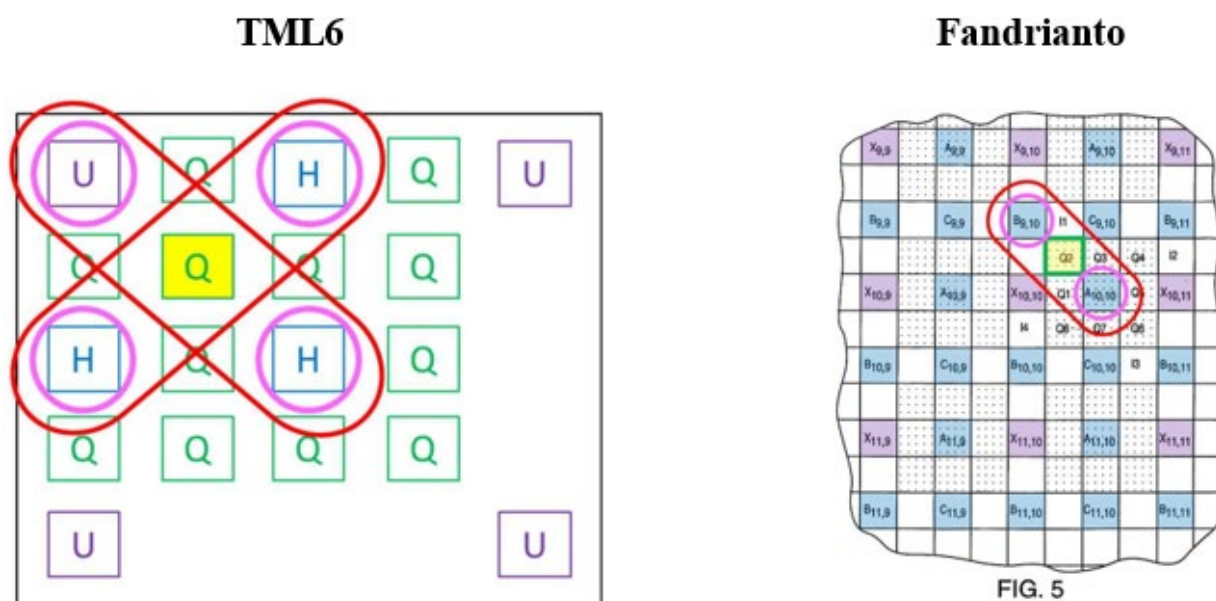
It would have been obvious to try Fandrianto's method to interpolate the quarter-quarter sub-pixels in TML6 because there are a finite number of identifiable and predictable ways to do so. *KSR*, 550 U.S. at 421. Interpolation refers generally to estimating new data points based on assumptions regarding surrounding known data points. *E.g.*, Britannica, 356 (“**interpolation**, in mathematics, the determination or estimation of the value of a function $f(x)$ from certain known values of the function.”). Sub-pixel interpolation is based on the corresponding assumption that values for sub-pixels at fractional locations are typically more similar to values of nearby pixels and sub-pixels than to those farther away. *E.g.*, VCEG-M28 (describing 2-tap, 4-tap, and 6-tap filters). Thus, a POSITA would have recognized there are a finite number of pixels and sub-pixels near the highlighted quarter-quarter sub-pixel (below) to reasonably use for interpolation—e.g., one adjacent unit pixel, and seven adjacent sub-pixels.



Ramchandran, ¶¶124-125. Accordingly, it would have been obvious to try a well-known one of these finite options—Fandrianto’s interpolation method for quarter-quarter sub-pixels—instead of TML6’s method. Ramchandran, ¶¶124-126.

Implementing Fandrianto’s method for interpolating quarter-quarter sub-pixels in TML6’s source code would have been a predictable result with a reasonable expectation of success. TML6 and Fandrianto disclose methods for interpolating quarter-quarter sub-pixels using the same inputs and basic arithmetic. For any quarter-quarter sub-pixel, Fandrianto’s interpolation method would achieve the same result in Fandrianto’s system as in the TML6 code. Ramchandran, ¶129.

Making the combination would have been trivial and thus predictable. The only difference between the TML6 and Fandrianto methods is TML6 discloses averaging both sets of two diagonally-opposed sub-pixels (red), whereas Fandrianto discloses averaging only one set of diagonally-opposed sub-pixels (also red).



In mathematical terms, Dr. Ramchandran explains the single diagonal interpolation of Fandrianto can be written as:

$$Q = \frac{H+H}{2}.$$

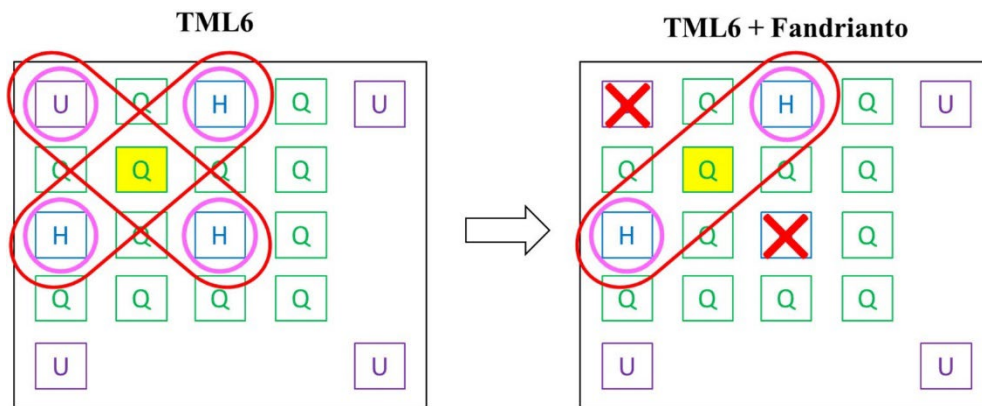
The average of TML6 can be written as:

$$Q = \frac{H+H+H+U}{4},$$

which is mathematically equivalent to averaging two diagonal interpolations:

$$Q = \frac{\left(\frac{H+H}{2}\right) + \left(\frac{H+U}{2}\right)}{2}.$$

Ramchandran, ¶130. TML6 therefore mathematically includes the average taught by Fandrianto, highlighted above, showing that TML6 teaches all of the required input values for such a calculation—i.e., values for the two half sub-pixels H. Accordingly, a POSITA would have easily combined Fandrianto and TML6 by either (1) using the interpolation hardware disclosed in Fandrianto’s Figure 2 (for which TML6 calculates all required inputs), or (2) removing one of the two diagonal interpolation calculations (shown above) for the quarter-quarter sub-pixels in TML6.



TML6, 58 (lines 593-625); Fandrianto, 10:51-58; Ramchandran, ¶¶130-131.

A POSITA would have expected predictable results and a reasonable likelihood of success because both TML6 and Fandrianto result in interpolated quarter sub-pixels, and as discussed above, TML6 discloses that the values required to interpolate the quarter sub-pixels of Fandrianto—unit pixels and half sub-pixels—are stored in the *imgY* and *mref* data structures. Moreover, implementing the quarter sub-pixel interpolation functionality in Fandrianto’s hardware, rather than TML6’s

software, would increase the speed with which quarter sub-pixels can be interpolated. Ramchandran, ¶132.

2. Claims 1, 4-17, 19-26, and 28-29

a. Independent Claim 1

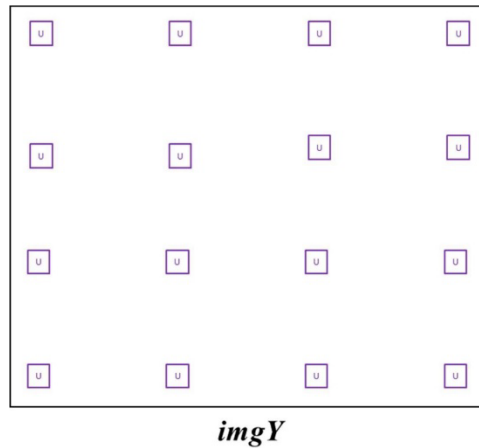
(i) 1[pre][a]

If limiting,⁸ TML6 teaches 1[pre][a]. Ramchandran, ¶¶133-134.

TML6 discloses “[a] method for sub-pixel value interpolation[.]” TML6, 48 (line 3). TML6 includes a function titled *encode_one_frame()* that “[e]ncodes one frame” of video data. TML6, 48-50 (lines 33-162). The *encode_one_frame()* function invokes functions *oneforthpix()* and *oneforthpix_2()* that interpolate sub-pixels for a given video frame. TML6, 49 (lines 102-109), 57-60 (lines 538-719); Ramchandran, ¶¶133-134, 47-108, 52 n.1.

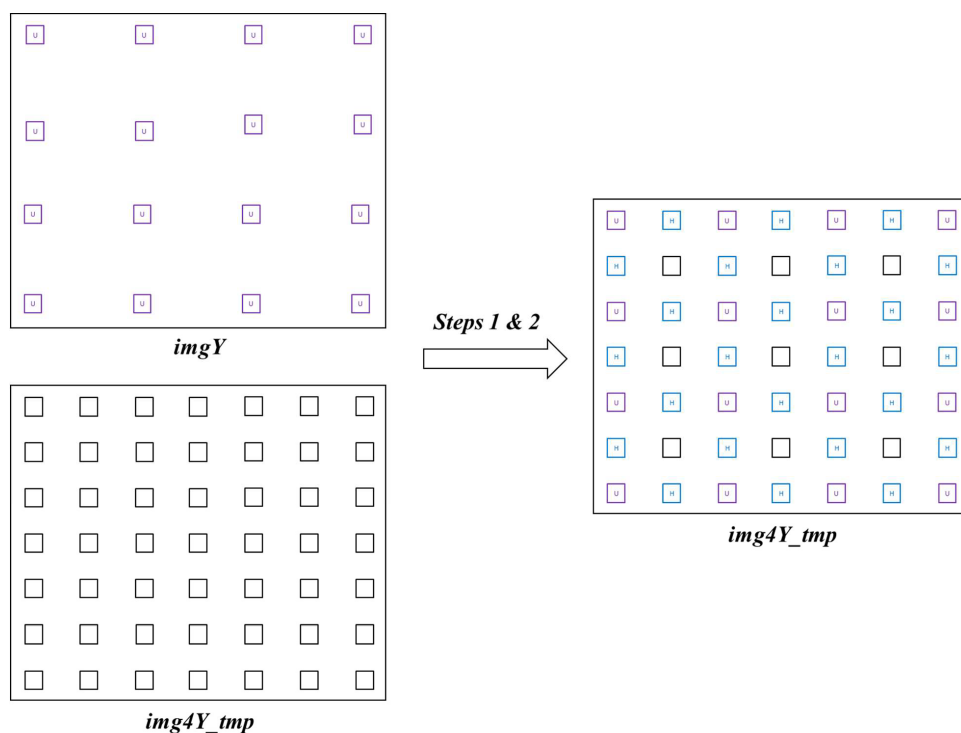
The *imgY* data structure, illustrated below, stores data values representing the brightness of unit pixels “U” (purple) for a video frame:

⁸ Petitioner reserves the right to argue in other proceedings the preamble of any challenged claim is limiting.

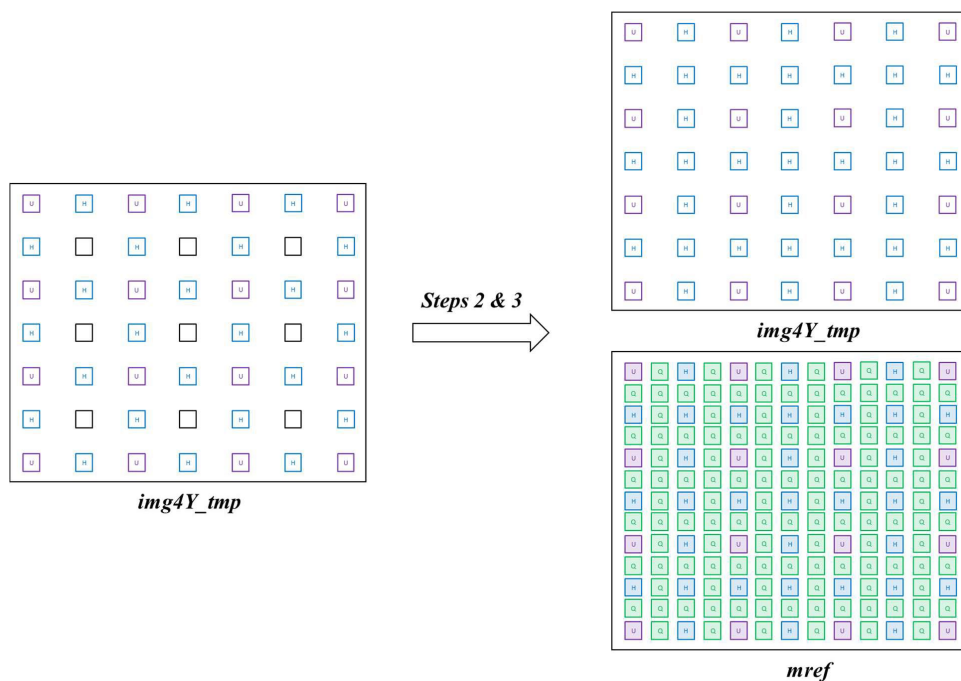


TML6, 93 (lines 1338-1344), 49-50 (lines 97-109), 110-112 (lines 480-600), 57-59 (lines 563-632), 131 (line 149).

The *oneforthpix()* function—described further *infra*, 1[a]-1[c]—performs three main steps to interpolate sub-pixel values based on the unit pixel values stored in *imgY*. TML6, 57-58 (lines 553-625); Ramchandran, ¶135. In the first two steps, *oneforthpix()* interpolates values for some of the half sub-pixels (blue) based on the values of the unit pixels (purple) and stores these values in *img4Y_tmp*.

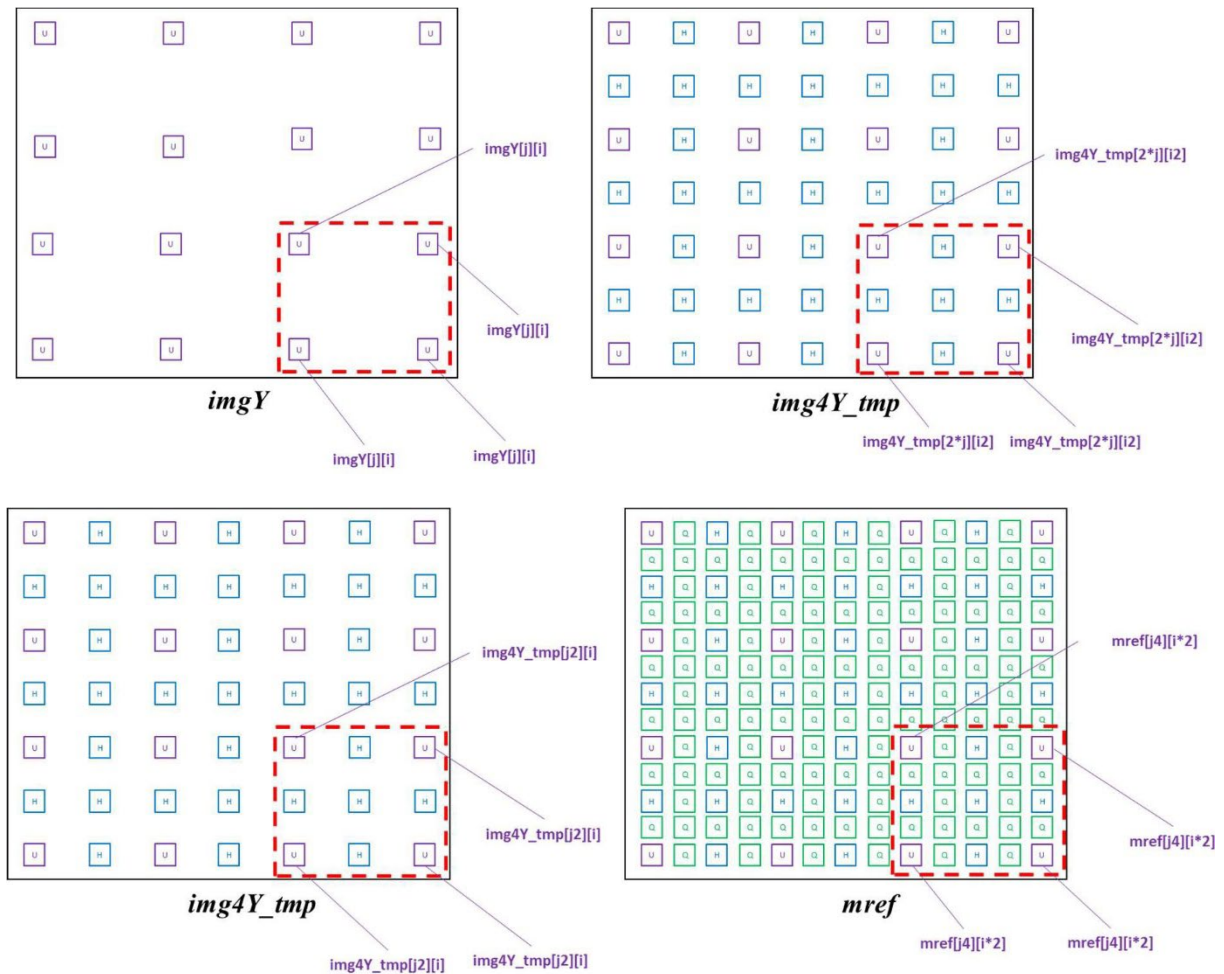


TML6, 57-58 (lines 563-592); Ramchandran, ¶136. In steps two and three, *oneforthpix()* interpolates additional half sub-pixels (blue) and stores these values in *img4Y_tmp*. The *oneforthpix()* function uses half sub-pixels (blue) and unit pixels (purple) in *img4Y_tmp* to interpolate values for quarter sub-pixels (green) and stores the quarter and half sub-pixel values (green) and (blue), in *mref*.



TML6, 57-58 (lines 576-625); Ramchandran, ¶137.

The *img4Y_tmp* and *mref* data structures include half sub-pixels (blue) and/or quarter sub-pixels (green) “situated within a rectangular bounded region [red] defined by four corner [unit] pixels [purple] with no intermediate pixels between the corners,” as recited. The pixels (purple) and sub-pixels (blue and green) are arranged “in rows and columns[.]”



TML6, 57-58 (lines 554-592). The pixels are “configured for display.” TML6, 1-2 (“picture formats”). Similarly, if a motion vector points to a sub-pixel, the sub-pixels are also “configured for display.” TML6, 57-59 (lines 538-632), 16-24 (lines 734-1180 (“motion_search” function)), 115-116 (lines 769-842 (“Motion vector search”)); Ramchandran, ¶138.

As illustrated below with reference to Figure 13(a), the '273 patent acknowledges TML6 discloses sub-pixel interpolation—interpolating half sub-pixels “b” and “c” (blue) and quarter sub-pixels “d” through “g” (green)—which are

situated within a rectangular bounded region (red) defined by four corner pixels “A” (purple), with no intermediate pixels (purple) between the corners, and with the pixels and sub-pixels arranged in rows and columns.

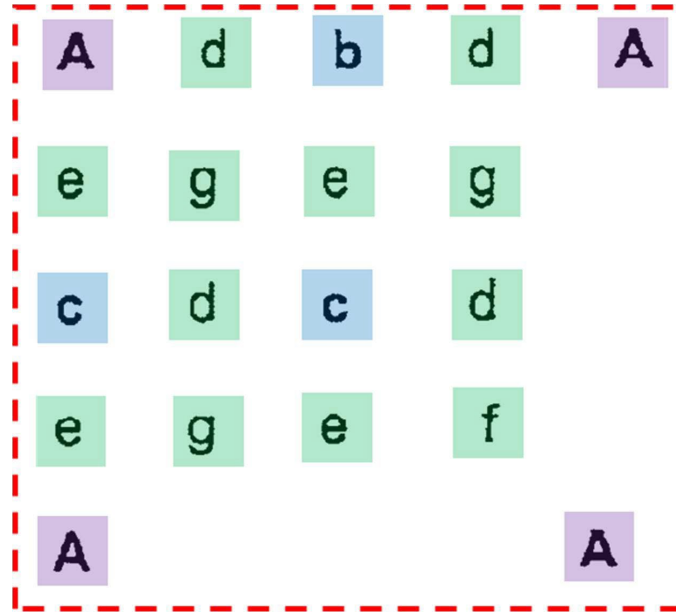


Fig. 13(a)

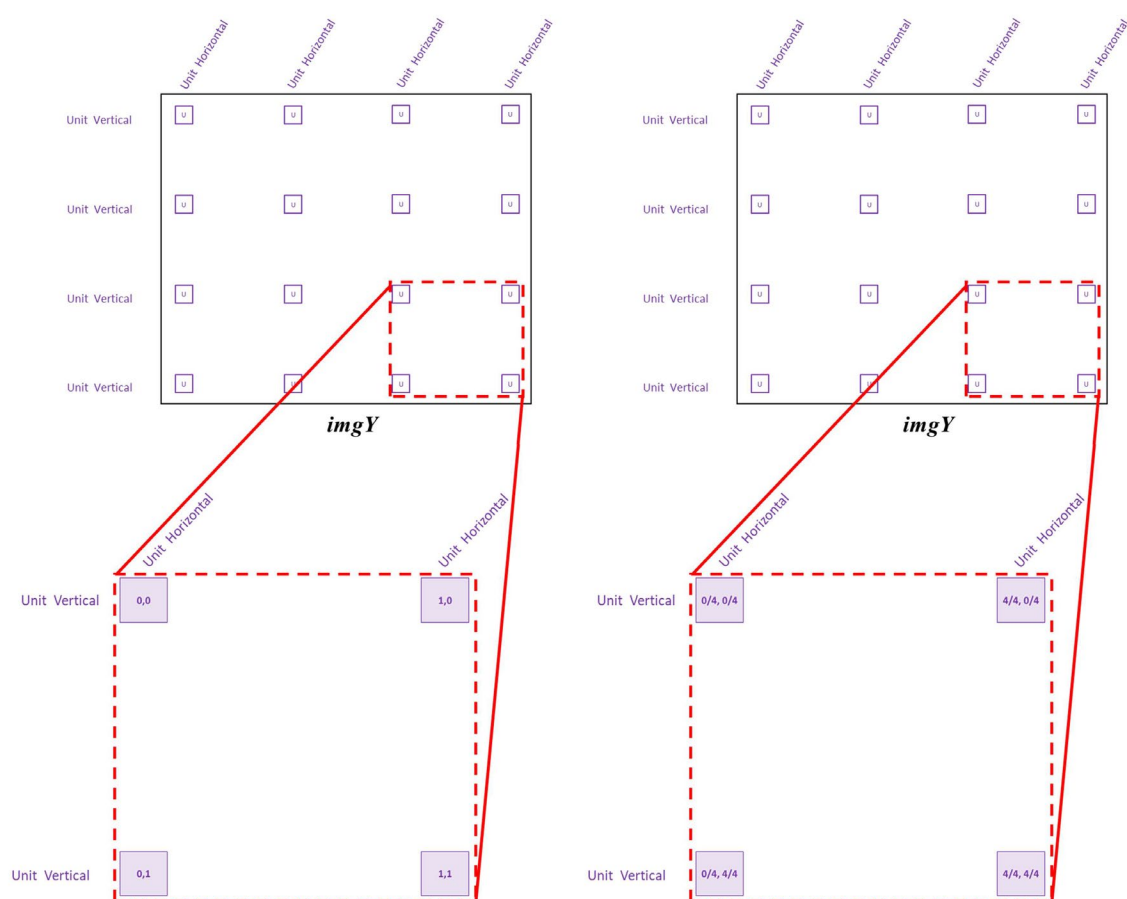
EX-1001, Figure 13(a), 8:38-9:36, 11:25-13:18; Ramchandran, ¶¶139-140.

TML6 thus interpolates sub-pixel values “situated within a rectangular bounded region” of a video frame, wherein the region is “defined by four corner pixels with no intermediate pixels [there]between..., the pixels and sub-pixels being configured for display in rows and columns,” as recited. Ramchandran, ¶¶133-140.

(ii) 1[pre][b]

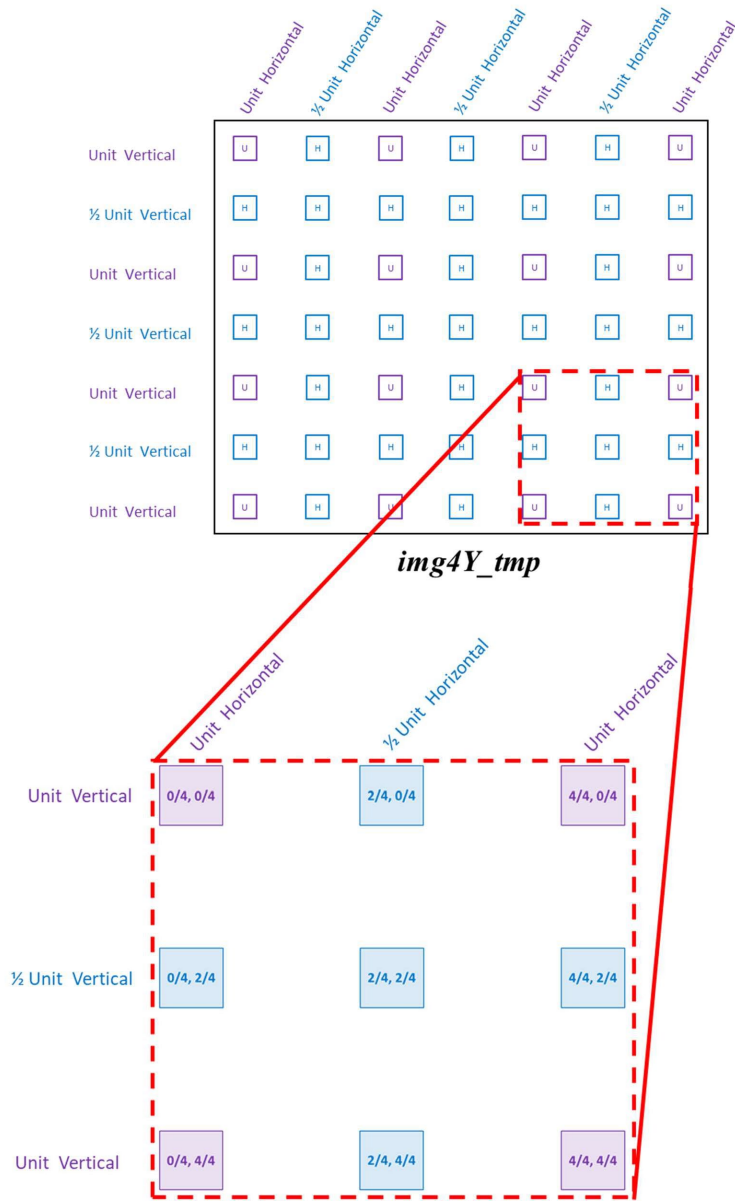
If limiting, TML6 teaches 1[pre][b]. Ramchandran, ¶¶141-142.

In TML6, the pixel/sub-pixel locations are representable mathematically within the rectangular bounded region. Ramchandran, ¶¶141-142. As discussed for 1[pre][a], *imgY* stores values for unit pixels “U” (purple) located in the rectangular bounded region (red; below left). The four unit pixels are representable mathematically using the Cartesian co-ordinate system (below right):



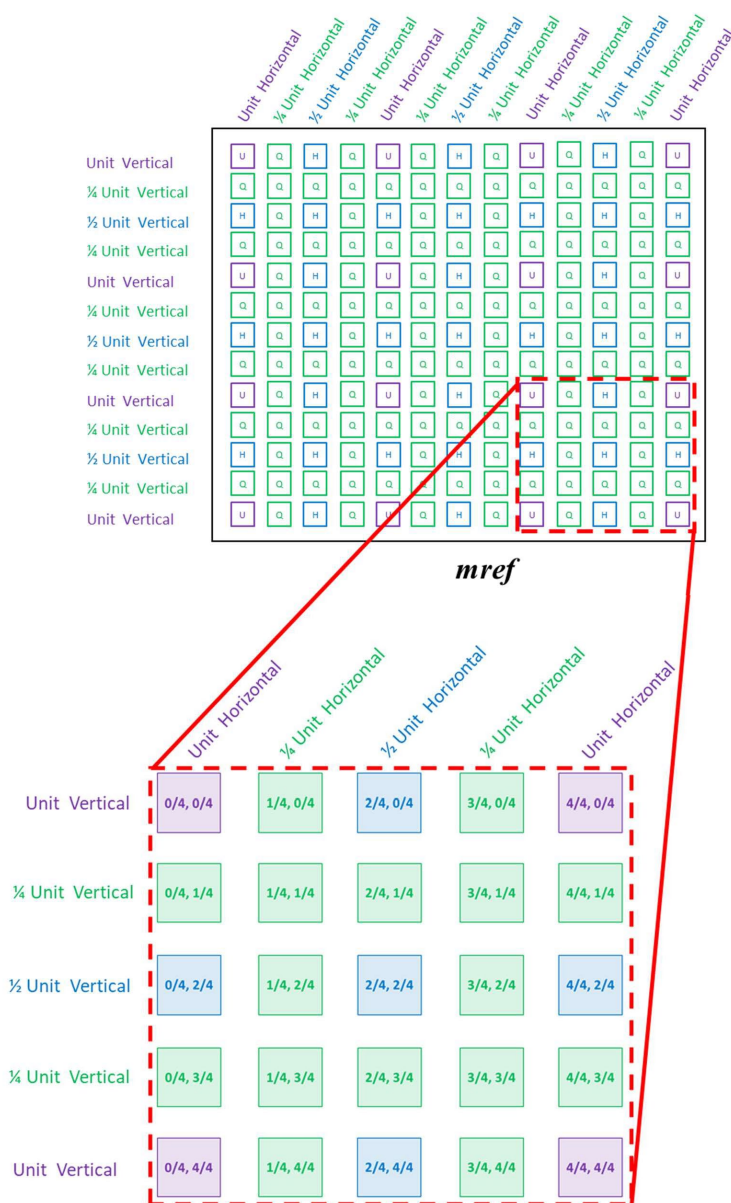
Ramchandran, ¶143.

The *oneforthpix()* function interpolates half sub-pixels (blue) based on the unit pixels U (purple) in *imgY* and stores these pixel and sub-pixel values in *img4Y_tmp*:



Ramchandran, ¶144.

The *oneforthpix()* function further interpolates quarter sub-pixels (green) based on the unit pixels U (purple) and the previously interpolated half sub-pixels (blue) and stores the pixel and sub-pixel values in *mref*:



Ramchandran, ¶145.

Accordingly, TML6 discloses locations of pixels (purple) and sub-pixels (blue and green) are representable mathematically within the rectangular bounded region (red) using the coordinate notation $\mathbf{K}/2^N$ for the horizontal co-ordinates and $\mathbf{L}/2^N$ for the vertical co-ordinates, as recited. The horizontal co-ordinates include: 0/4 ($\mathbf{K}=0$),

1/4 (**K=1**), 2/4 (**K=2**), 3/4 (**K=3**), and 4/4 (**K=4**); and the vertical co-ordinates include: 0/4 (**L=0**), 1/4 (**L=1**), 2/4 (**L=2**), 3/4 (**L=3**), and 4/4 (**L=4**) when **N=2**. **K** and **L** are thus positive integers between zero and 2^N (**2²=4**), as recited.⁹ Ramchandran, ¶¶146-147.

As discussed *supra* §VII.A, TML6 interpolates up to 1/4 ($=1/2^2$) sub-pixels. Therefore, TML6 teaches **N=2**, “a positive integer greater than one...representing a particular degree of sub-pixel value interpolation,” as recited. Ramchandran, ¶148.

The '273 patent confirms, with reference to Figure 13(a), that TML6 discloses locations of pixels (purple), half sub-pixels (blue), and quarter sub-pixels (green) representable mathematically within the rectangular bounded region (red) using the coordinate notation $K/2^N, L/2^N$ as claimed.

⁹ Claim 1 recites **K** and **L** can be zero yet also describes them as “**positive integers**...between zero and 2^N .” Petitioner assumes **K** and **L** can be zero for this proceeding but reserves the right to argue Claim 1 is invalid as indefinite in other proceedings.

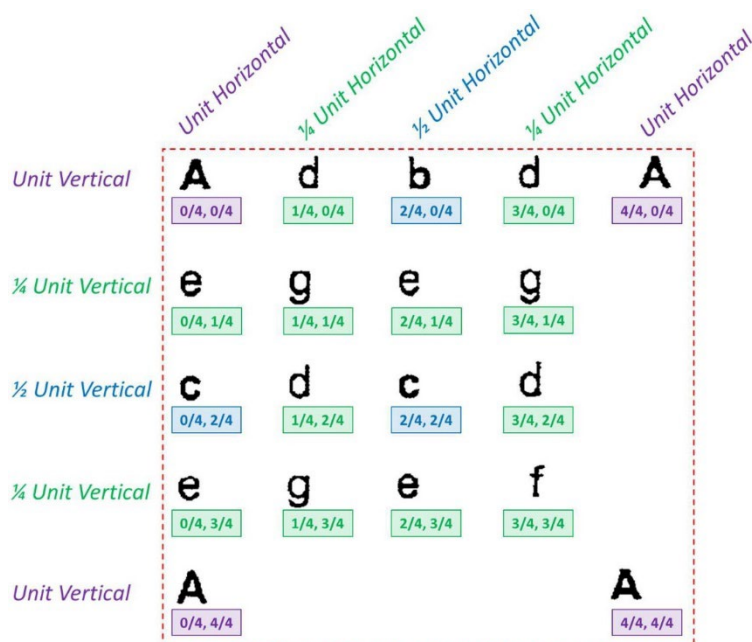


Fig. 13(a)

EX-1001, Figure 13(a), 8:38-9:53, 11:25-13:18; Ramchandran, ¶149.

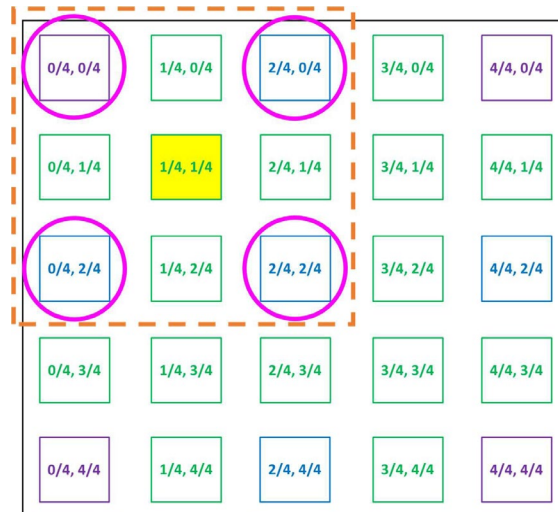
Finally, TML6 is source code a POSITA would have understood is designed to cause a processor, such as a GPU, to execute commands. *See* TML6. TML6 therefore discloses causing an apparatus to perform the interpolation steps. Ramchandran, ¶148.

TML6 thus discloses representing pixels/sub-pixels “within the rectangular bounded region” of 1[pre][a] “mathematically...using...co-ordinate notation $K/2^N$, $L/2^N$,” where $N=2$ (“a positive integer greater than one and representing a particular degree of sub-pixel value interpolation”) and K and L equal 0 to 4 (“positive integers having respective values between zero and 2^N ”), as recited. Ramchandran, ¶¶141-149.

(iii) 1[a]

The TML6-Fandrianto combination teaches 1[a]. Ramchandran, ¶150.

As illustrated below, TML6's *oneforthpix()* function interpolates the highlighted quarter sub-pixels situated at the co-ordinates $(1/4, 1/4)$, i.e., $K/2^N, L/2^N$ ($K=1, L=1, N=2$). These $(1/4, 1/4)$ sub-pixels are interpolated according to a weighted average—weights equal to “1”—of the four values (circled) comprising one pixel (purple) and three half sub-pixels (blue).



TML6, 58 (lines 593-625); Ramchandran, ¶¶150-151. As shown above, these pixel/sub-pixels are situated within a quadrant (orange) of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $(1/2, 1/2)$ — $(2/4, 2/4)$ ¹⁰ sub-pixel (blue; circled bottom right)—and the nearest-neighbouring

¹⁰ $(1/2, 1/2)$ and $(2/4, 2/4)$ refer to the same sub-pixel in half- and quarter-pixel resolution, respectively. Ramchandran, ¶151.

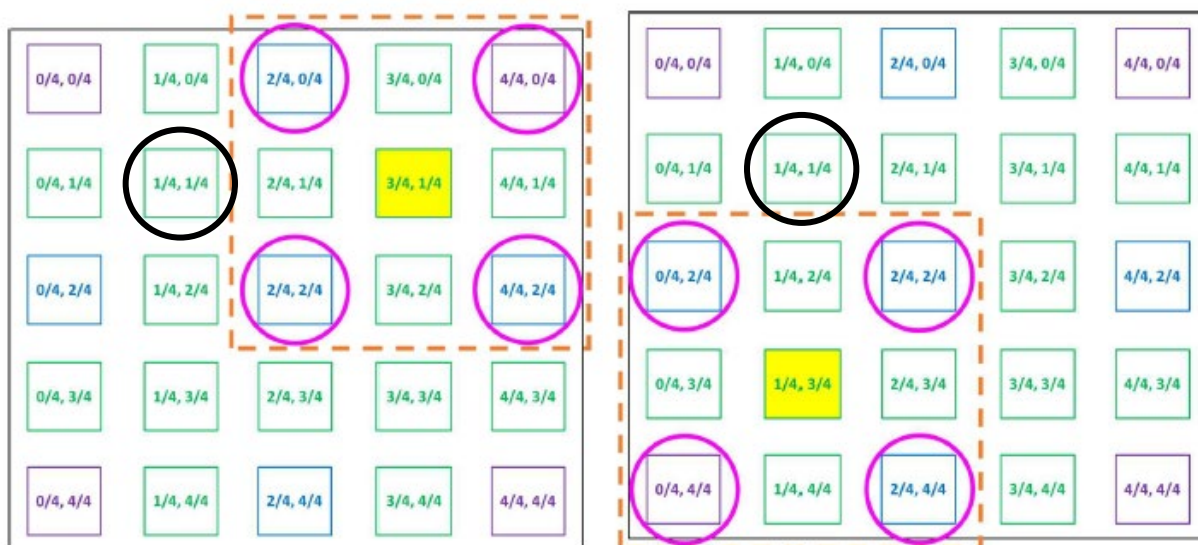
pixel—(0/4,0/4) pixel (purple; circled top left). The '273 patent confirms TML6 interpolates (1/4,1/4) sub-pixels using a weighted average of the aforementioned unit pixel and three half sub-pixels. EX-1001, Figure 13(a), 12:34-40; Ramchandran, ¶153.

The following chart maps each pixel/sub-pixel to the corresponding element of 1[a]:

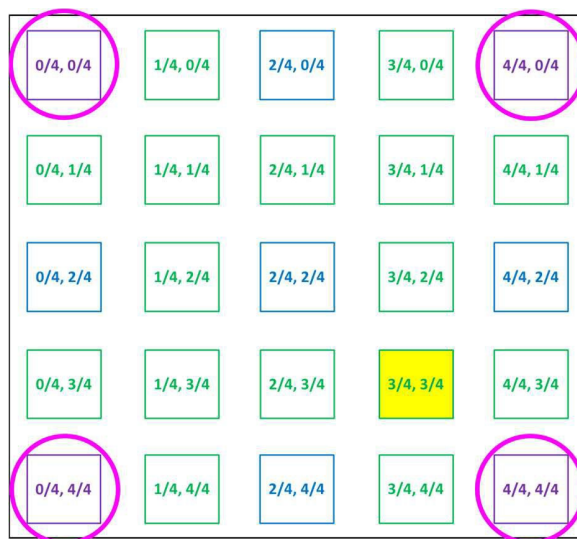
Pixel/sub-pixel	1[a] element
(0/4,0/4) pixel	“nearest-neighbouring pixel”
(2/4,2/4) (1/2,1/2) sub-pixel	“sub-pixel situated at co-ordinates $\frac{1}{2}, \frac{1}{2}$ ”
(0/4,2/4) K=0; L=2; N=2	“a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, situated within a quadrant of the rectangular bounded region, the quadrant being defined by the sub-pixel having co-ordinates $\frac{1}{2}, \frac{1}{2}$ and the nearest neighbouring pixel”
(2/4,0/4) K=2; L=0; N=2	

Ramchandran, ¶¶150-151, 47-108.

Similarly, as shown below, the highlighted sub-pixel on the right situated at the co-ordinates (1/4,3/4)— $K/2^N, L/2^N$ (**K=1, L=3, N=2**)—and the highlighted sub-pixel on the left situated at the co-ordinates (3/4,1/4)— $K/2^N, L/2^N$ (**K=3, L=1, N=2**)—along with corresponding quarter sub-pixels in other rectangular bounded regions, are interpolated in the same manner as the (1/4,1/4) sub-pixels (circled; black).



TML6, 58-59 (lines 593-632). Finally, as shown below, the highlighted (3/4,3/4) sub-pixel is interpolated using the nearest four unit pixels (purple).



Id.; Ramchandran, ¶152.

TML6 therefore interpolates the highlighted (1/4,1/4), (1/4,3/4), and (3/4,1/4) sub-pixels in the above figures using a weighted average of either (1) “the value of a nearest-neighbouring pixel,” (2) “the value of the sub-pixel situated at co-ordinates

$\frac{1}{2}, \frac{1}{2}$,” or (3) “the values of a pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero.” If TML6 does not teach interpolating the highlighted $(\frac{1}{4}, \frac{1}{4})$, $(\frac{1}{4}, \frac{3}{4})$, $(\frac{3}{4}, \frac{1}{4})$, or $(\frac{3}{4}, \frac{3}{4})$ sub-pixel according to a predetermined choice of either (1) a weighted average of the values of a pair of the nearest-neighbouring pixel and the sub-pixel situated at co-ordinates $(\frac{1}{2}, \frac{1}{2})$, or (2) a weighted average of the values of the pair of diagonally-opposed sub-pixels having co-ordinates with even values of K and L, including zero, the combination of TML6 and Fandrianto does. Ramchandran, ¶154.

Fandrianto, in Figure 5 (below), shows four unit pixels X (purple) within a rectangular bounded region (red) and interpolates highlighted $(\frac{1}{4}, \frac{3}{4})$ sub-pixel Q2 (green) having co-ordinates $(K/2^N, L/2^N)$ with odd values of K and L ($K=1$, $L=3$, $N=2$). Sub-pixel Q2 is interpolated using a weighted average of the pair of diagonally-opposed sub-pixels at co-ordinates $(0/4, 2/4)$ ($B_{9,10}$) and $(2/4, 4/4)$ ($A_{10,10}$) (both blue; circled).¹¹ The $(0/4, 2/4)$ sub-pixel $B_{9,10}$ and the $(2/4, 4/4)$ sub-pixel $A_{10,10}$

¹¹ As explained *supra* §VII.B, A block 72 and B block 74 contain “A” and “B” data, respectively, because Fandrianto does not require a specific order for deriving pixels Q1-Q8. Ramchandran, ¶¶156-158 (citing Fandrianto, 7:26-8:20, 10:51-58, Figures 3B, 5).

have co-ordinates $(K/2^N, L/2^N)$ with even values of K and L (**K=0 or 2, L=2 or 4, N=2**) including zero.

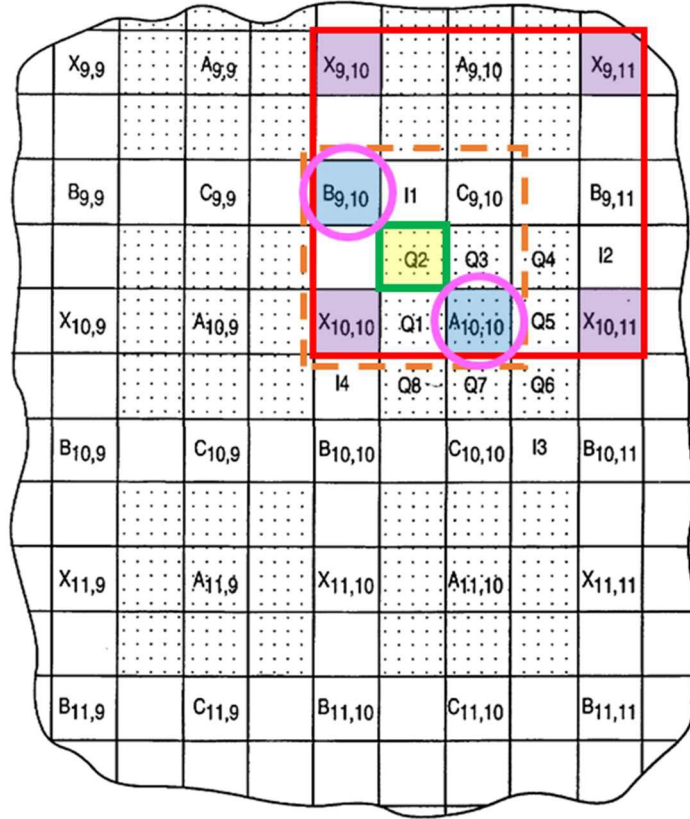


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:58-9:32, 10:51-58. As shown above, these diagonally-opposed $(0/4, 2/4)$ and $(2/4, 4/4)$ sub-pixels are situated within a quadrant (orange box) of the rectangular bounded region defined by the sub-pixel having co-ordinates $(1/2, 1/2)$ — $(2/4, 2/4)$ sub-pixel $C_{9,10}$ —and the nearest-neighbouring pixel— $(0/4, 4/4)$ pixel $X_{10,10}$. *Id.*; Ramchandran, ¶160.

Fandrianto further teaches interpolating the highlighted $(1/4, 3/4)$ sub-pixel Q2 using a weighted average of the value of a nearest-neighbouring pixel— $(0/4, 4/4)$

pixel $X_{10,10}$ (purple; circled)—and the value of the sub-pixel $C_{9,10}$ situated at co-ordinates $(1/2, 1/2)$ (blue; circled). Fandrianto explains “when **diagonal interpolation** is...used, all quarter pixel estimation search blocks [e.g., sub-pixel Q2 (green)] are interpolated from the full [pixel X (purple)] and half pixel [C (blue)] estimation search blocks.”

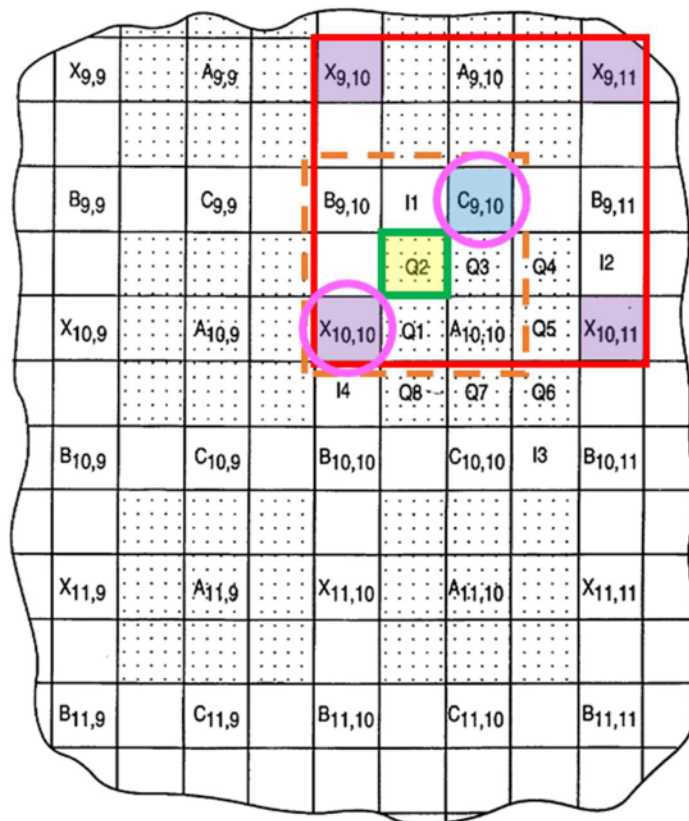


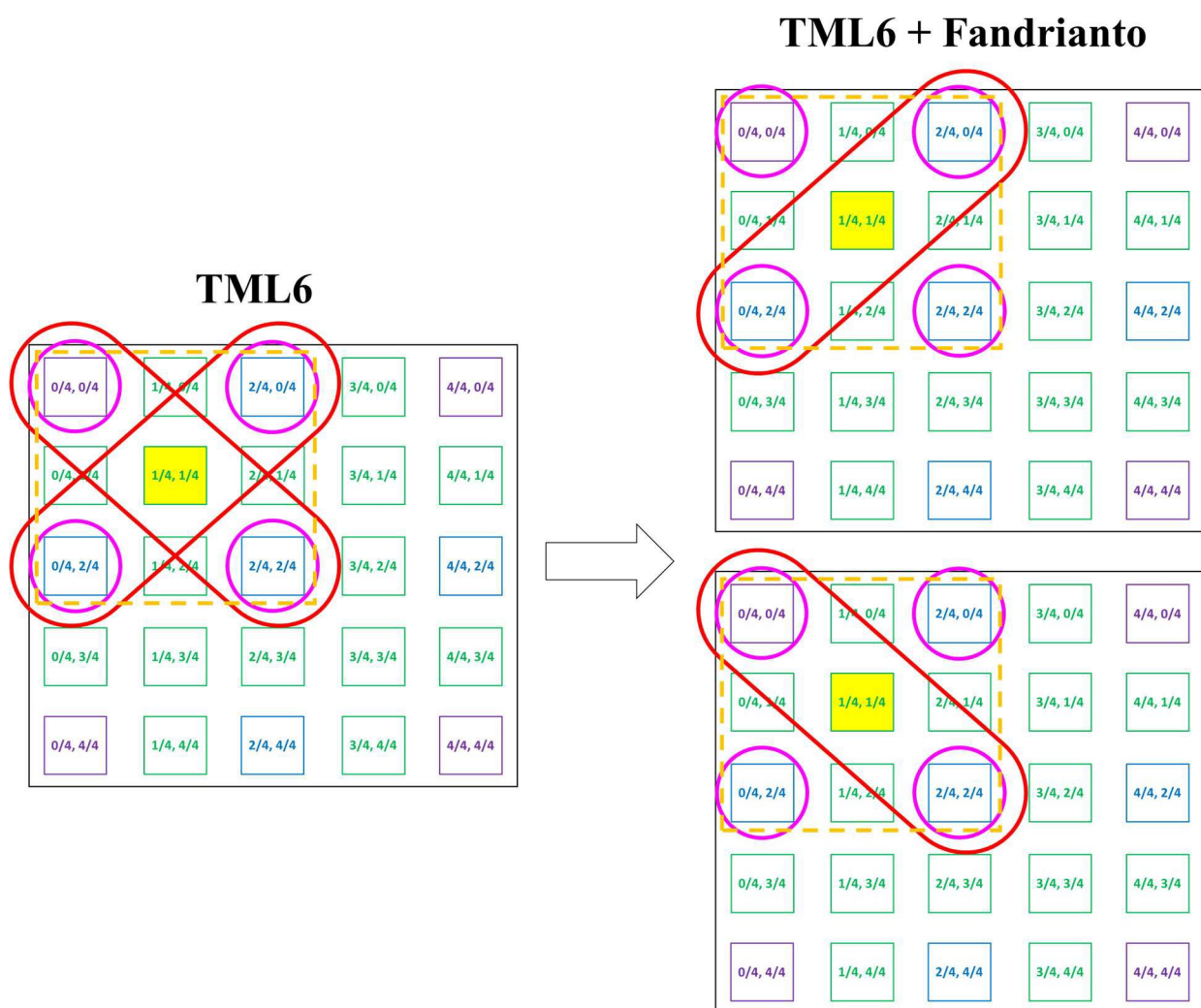
FIG. 5

Fandrianto, Figure 5, 5:26-32, 7:52-55, 8:58-9:32, 10:51-58. Fandrianto therefore diagonally interpolates $(1/4, 3/4)$ sub-pixel Q2 using the nearest-neighbouring pixel $X_{10,10}$ and the sub-pixel $C_{9,10}$ with co-ordinates $(1/2, 1/2)$, as recited. *Id.* Like other

disclosed interpolations, this interpolation is a weighted average with weights equal to “1.” Fandrianto, Figure 5, 6:18-7:2, 8:35-10:50; Ramchandran, ¶¶160-161.

As discussed *supra* §VIII.A.1, a POSITA would have been motivated to implement Fandrianto’s method of interpolating the $(1/4, 3/4)$ sub-pixel Q2 (green) in TML6 to interpolate the sub-pixels at co-ordinates $(1/4, 1/4)$, $(1/4, 3/4)$, $(3/4, 1/4)$, and/or $(3/4, 3/4)$. A POSITA would have done so by modifying TML6 to use Fandrianto’s interpolation method or using Fandrianto’s technique on its hardware. Thus, as shown below, the combined teachings of TML6 and Fandrianto disclose interpolating TML6’s $(1/4, 1/4)$ sub-pixel (green) using either:

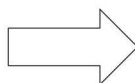
- a weighted average of a nearest-neighbouring $(0/4, 0/4)$ pixel (purple) and the $(2/4, 2/4)$ sub-pixel (blue) situated at co-ordinates $(1/2, 1/2)$ (bottom right), or
- a weighted average of a pair of diagonally-opposed sub-pixels— $(0/4, 2/4)$ and $(2/4, 0/4)$ sub-pixels (blue) having co-ordinates with even values of K and L, including zero ($K=0$ or 2 , $L=0$ or 2)—situated within a quadrant (orange box) of the rectangular bounded region defined by the sub-pixel having co-ordinates $(1/2, 1/2)$ and the nearest-neighbouring pixel (upper right), as in Fandrianto.

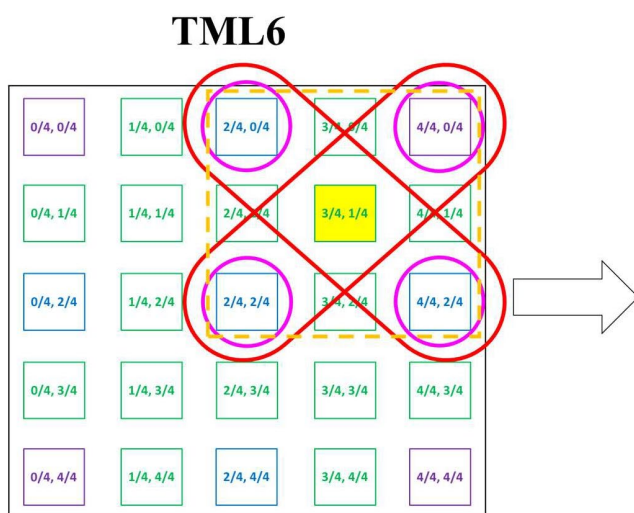
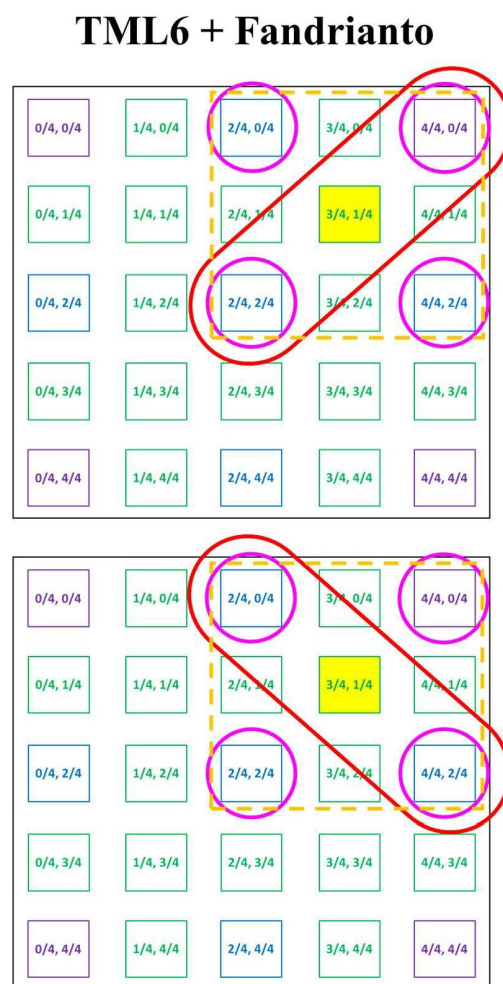


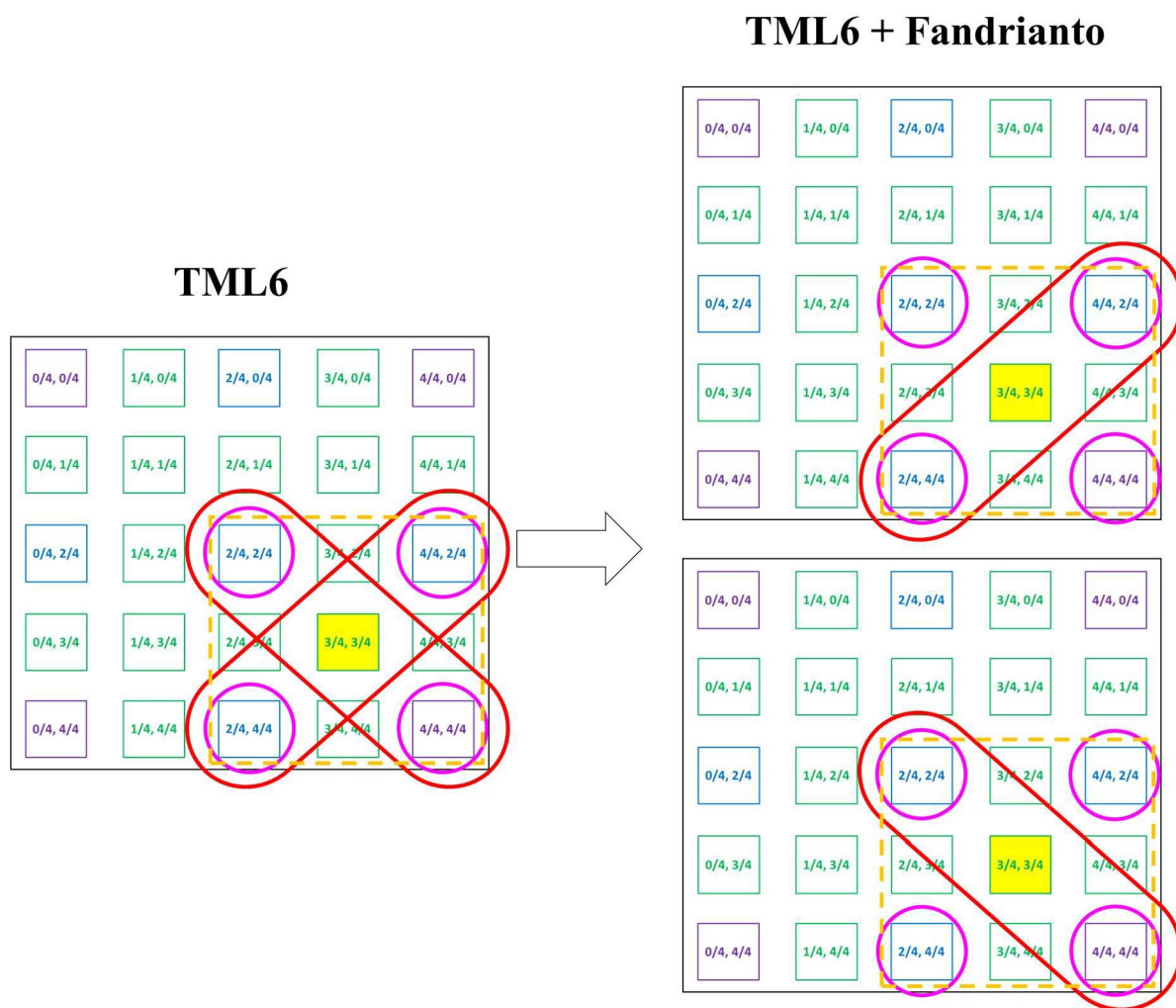
As shown below, the TML6-Fandrianto combination interpolates TML6's (1/4,3/4), (3/4,1/4), and (3/4,3/4) sub-pixels (green) the same way.

TML6 + Fandrianto

TML6







Id.; Ramchandran, ¶162.

As discussed *supra* §VIII.A.1, it would have been obvious for a POSITA to interpolate TML6's (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4) sub-pixels using a predetermined choice of either of the two diagonal interpolation methods of Fandrianto. Fandrianto explicitly teaches a finite number of interpolation techniques for the (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4) sub-pixels, including vertical interpolation and the two diagonal interpolations. Thus, it would have been obvious

to try these finite options, including making a predetermined choice between either of the two diagonal interpolation options, to interpolate any of the (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4) sub-pixels. Revising TML6 to implement this modification or using Fandrianto's hardware would have been straightforward, and a POSITA would have reasonably expected this modification to yield the predictable result of interpolating (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4) sub-pixels based on values of nearby pixels and/or sub-pixels. Ramchandran, ¶163.

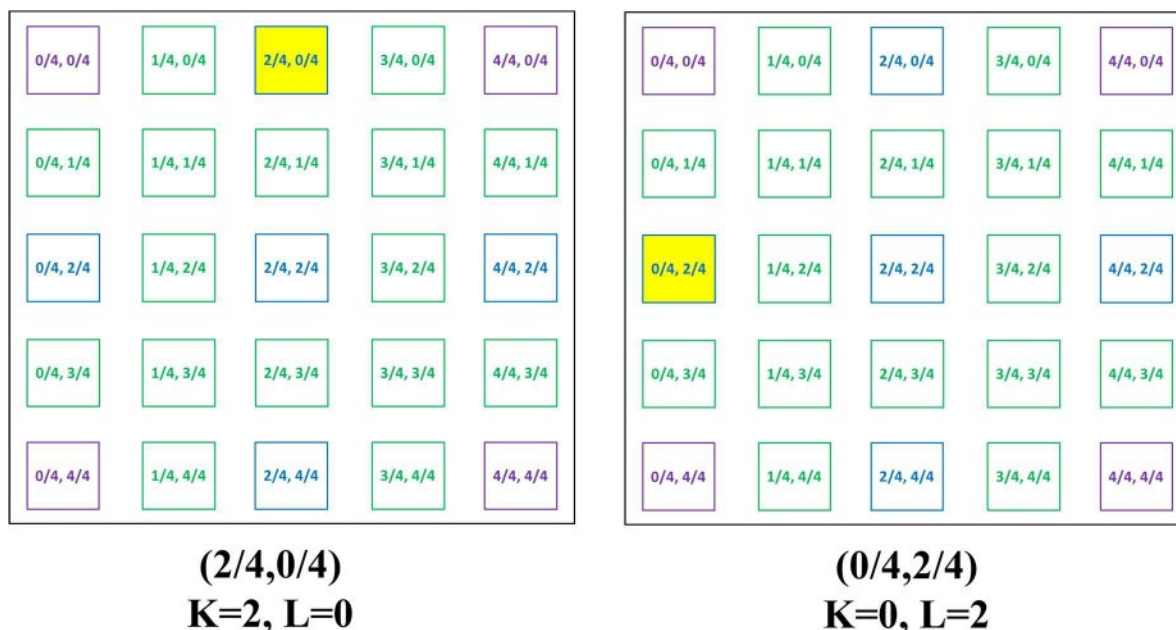
The TML6-Fandrianto combination thus teaches “interpolat[ing] a...value for a sub-pixel having co-ordinates with odd values of K and L,” i.e., K=1, L=3, “according to a predetermined choice of either [of] a [pair of] weighted average[s]” calculated using the values mapped in the table above and in the manner described above, as recited. Ramchandran, ¶163.

(iv) 1[b]

TML6 teaches 1[b]. Ramchandran, ¶164.

The *oneforthpix()* function interpolates values for the (2/4,0/4) sub-pixels “having co-ordinates $[K/2^N, L/2^N]$ with K equal to an even value and L equal to zero

[**K=2, L=0, N=2**]” and for the (0/4,2/4) sub-pixels “having co-ordinates [$K/2^N, L/2^N$] with K equal to zero and L equal to an even value [**K=0, L=2, N=2**],” as recited.¹²

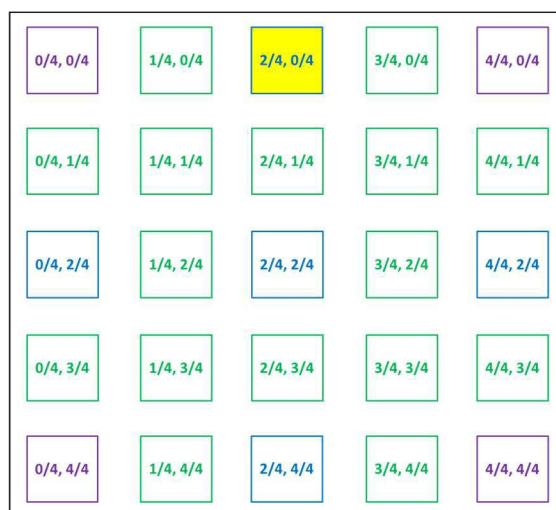


TML6, 57-58 (lines 538-592). As discussed for 1[a], these (2/4,0/4) and (0/4,2/4) sub-pixels are used to interpolate sub-pixels having co-ordinates (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4)—with odd values (1 or 3) of K and L. The *oneforthpixmap* function interpolates values for the (2/4,0/4) and (0/4,2/4) sub-pixels using weighted

¹² If Claim 1 requires interpolating, in one rectangular bounded region, multiple sub-pixels having the recited co-ordinates, Petitioner reserves the right to argue that the claim is invalid as indefinite in other proceedings, since there is only one such sub-pixel when N=2. Ramchandran, ¶164 n.15.

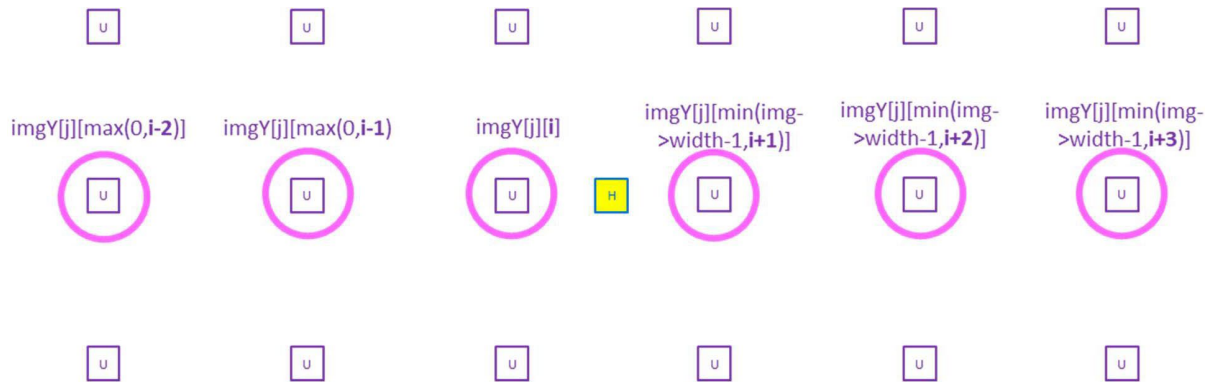
sums of the values of the unit pixels (purple), which are situated in rows and columns. TML6, 57-58 (lines 538-592); Ramchandran, ¶¶164, 47-108.

The *oneforthpix()* function interpolates highlighted (2/4,0/4) sub-pixels (blue; below) using values in *imgY* of unit pixels (purple) in the same row as the highlighted sub-pixel.



(2/4,0/4)
K=2, L=0

These highlighted (2/4,0/4) sub-pixels (blue) are interpolated in the first step of the *oneforthpix()* function using a weighted sum of the values of the three adjacent circled unit pixels (purple) to the left and the three adjacent circled unit pixels (purple) to the right.



Ramchandran, ¶165 (discussing TML6, 57-58 (lines 563-575, 588)).

The weights given to the value of each unit pixel (purple) for this weighted sum are stored in the **ONE_FOURTH_TAP** data structure.¹³ TML6, 57 (lines 568-570), 3 (lines 28-34). A value including this weighted sum is stored in *img4Y_tmp* and then, after being rounded and brought into the dynamic range, is stored in *mref*. TML6, 57-58 (lines 572, 588); Ramchandran, ¶165.

The '273 patent's Figure 13(b) (below) confirms TML6 interpolates (2/4,0/4) sub-pixel "b" (blue) "based upon the values of the 6 pixels (A₁ to A₆) [purple] situated in a row at unit horizontal locations and unit vertical locations symmetrically about b[.]"

¹³ The weighted sum—each of the weights from **ONE_FOURTH_TAP**—is multiplied by 32. TML6, 57 (line 572).

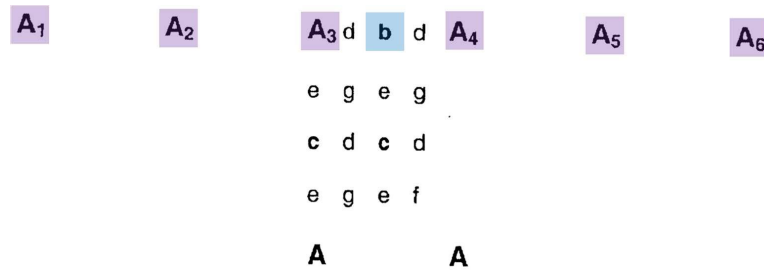
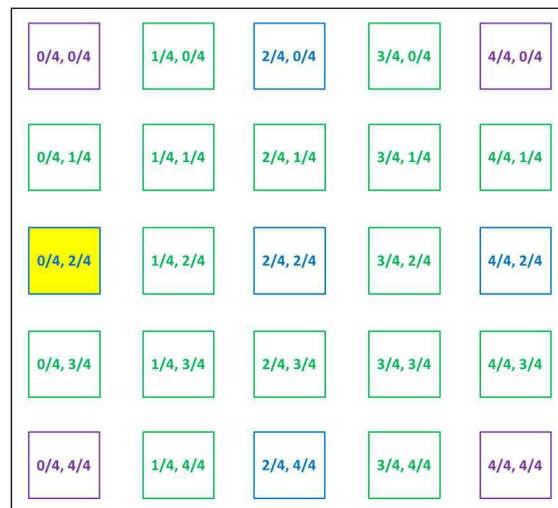


Figure13(b)

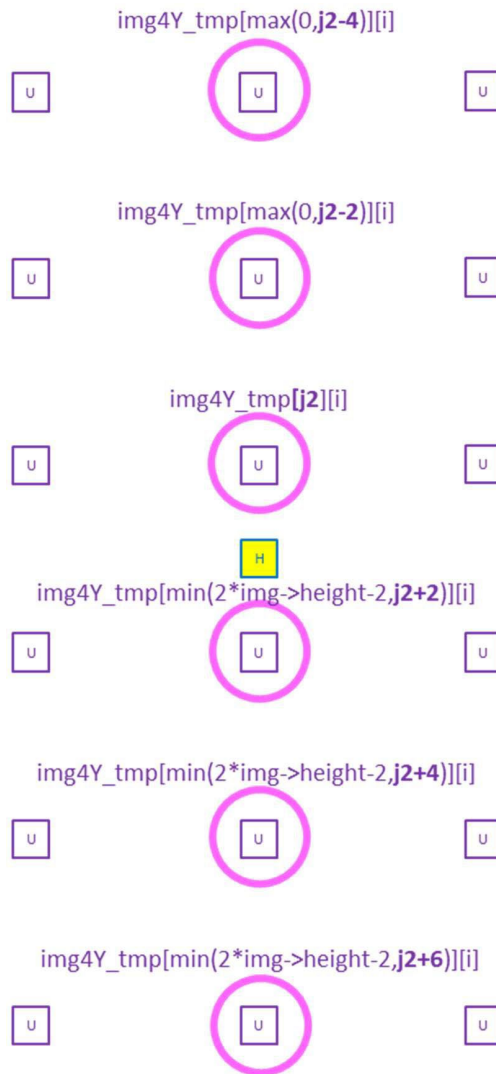
EX-1001, Figure 13(b), 11:35-45; Ramchandran, ¶166.

Similarly, *oneforthpix()* interpolates highlighted (0/4,2/4) sub-pixels (blue; below) using unit pixels (purple) in the same column as the highlighted sub-pixels.



(0/4,2/4)
K=0, L=2

These highlighted (0/4,2/4) sub-pixels are interpolated in *oneforthpix()*'s second step using a weighted sum of the values of the three adjacent circled unit pixels (purple) above and the three adjacent circled unit pixels (purple) below.



Ramchandran, ¶167 (discussing TML6, 57-58 (lines 576-592)).

The weights for this weighted sum are stored in **ONE_FOURTH_TAP**, discussed above. TML6, 57-58 (lines 576-592), 3 (lines 28-34). A value including this weighted sum is stored in **img4Y_tmp** and then, after being rounded and brought into the dynamic range, is stored in **mref**. TML6 57-58 (lines 576-592); Ramchandran, ¶168.

Figure 13(c) of the '273 patent, below, confirms how TML6 interpolates these half sub-pixels. The value for the (0/4,2/4) sub-pixel “c” (blue) is interpolated “based upon the values of the 6 pixels (A_1 to A_6) [purple] situated in a column at unit horizontal locations and unit vertical locations symmetrically about c, according to the formula $c=(A_1-5A_2+20A_3+20A_4-5A_5+A_6)$.”

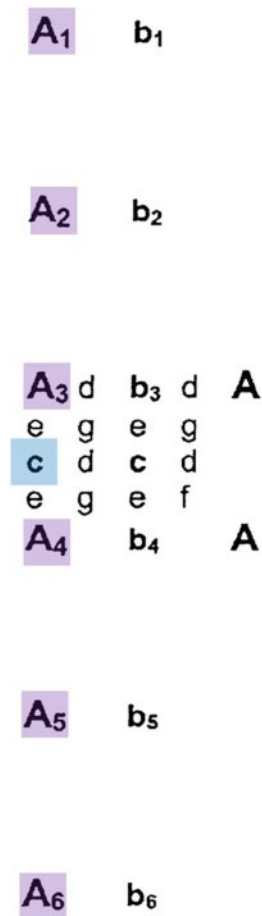


Figure 13(c)

EX-1001, Figure 13(c), 11:46-11:63; Ramchandran, ¶169.

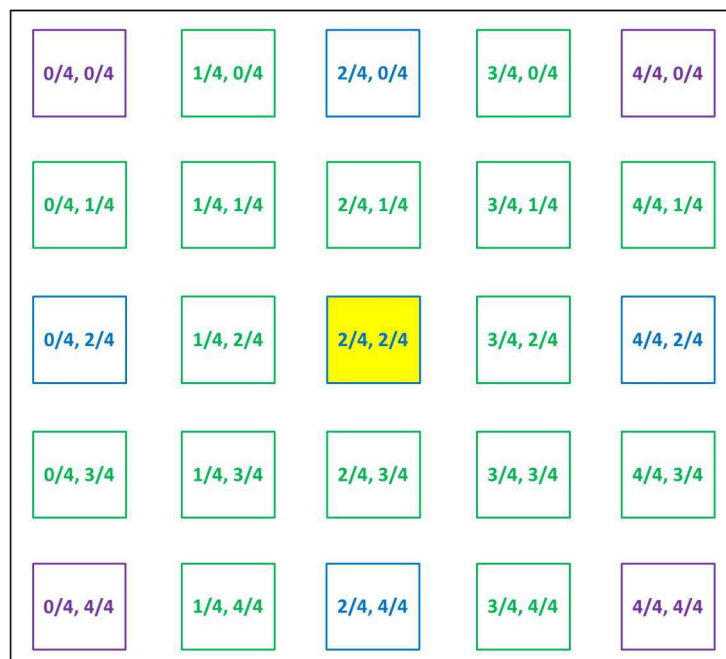
TML6 thus interpolates (2/4,0/4) sub-pixels (**K=2, L=0, N=2**; “sub-pixels having co-ordinates with K equal to an even value and L equal to zero”) and (0/4,2/4) sub-pixels (**K=0, L=2, N=2**; “sub-pixels having co-ordinates with K equal to zero and L equal to an even value”) for “use[] in the interpolation of” (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4) sub-pixels (**K=1 or 3, L=1 or 3, N=2**; “sub-pixels having co-ordinates with odd values of K and L”) “using weighted sums of the [pixel] values” derived from weights stored in **ONE_FOURTH_TAP**, as recited. Ramchandran, ¶164.

(v) 1[c]

The TML6-Fandrianto combination teaches 1[c]. Ramchandran, ¶170.

As shown below, TML6’s *oneforthpix()* function interpolates values for (2/4,2/4) sub-pixels (blue) (highlighted) “having co-ordinates with even values of K and L” ($K/2^N, L/2^N$; **K=L=N=2**).¹⁴

¹⁴ If Patent Owner argues that 1[c] requires interpolating multiple sub-pixels in one rectangular bounded region with even values of K and L, Petitioner reserves the right to argue that the claim is invalid as indefinite in other proceedings since there is only one such sub-pixel when N=2. Ramchandran, ¶170 n.16.



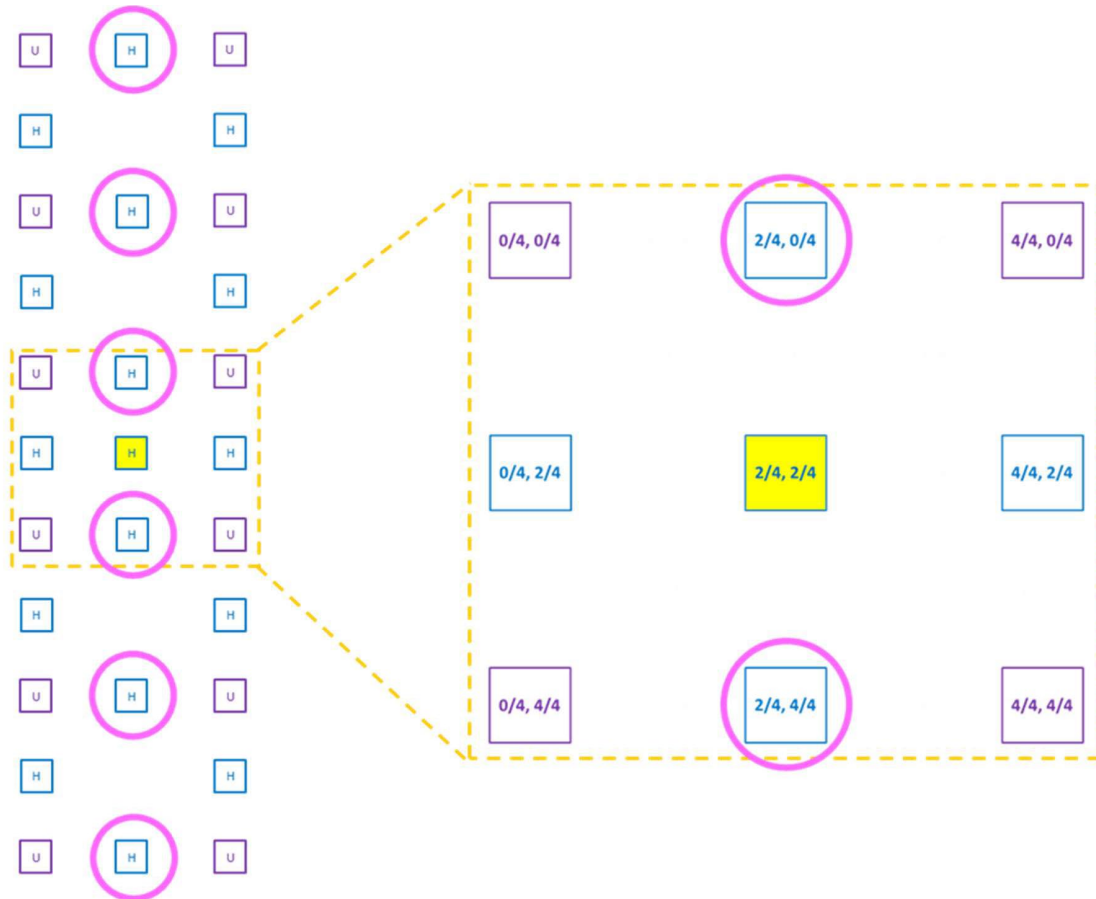
(2/4,2/4)
K=2, L=2

TML6, 57-58 (lines 576-592). As discussed for 1[a], these (2/4,2/4) sub-pixels are “situated at co-ordinates $\frac{1}{2}, \frac{1}{2}$ ” and used in the interpolation of (1/4,1/4), (1/4,3/4), (3/4,1/4), and/or (3/4,3/4) sub-pixels “having co-ordinates with odd values of K and L” ($K/2^N, L/2^N$; **K=1 or 3, L=1 or 3, N=2**), as recited. Ramchandran, ¶¶170, 47-108.

The highlighted (2/4,2/4) sub-pixels are interpolated in the second step of *oneforthpix()* using a weighted sum (hereinafter, “first weighted sum”) of:

- The (2/4,0/4) sub-pixel and (2/4,4/4) sub-pixel (blue; circled) (in the orange box) having co-ordinates with K equal to an even value and L equal to zero ($K/2^N, L/2^N$; **K=2, L=0 or 4, N=2**), and

- The circled $(2/4, 0/4)$ sub-pixel and $(2/4, 4/4)$ sub-pixel having corresponding co-ordinates in immediately adjacent rectangular bounded regions.



First Weighted Sum

TML6, 57-58 (lines 576-592). A value with this first weighted sum is stored in *img4Y_tmp* and then, after being rounded and brought into the dynamic range, stored in *mref*. TML6, 57-58 (lines 576-592); Ramchandran, ¶171.

Figure 13(c) of the '273 patent (below) confirms how TML6 interpolates the $(2/4, 2/4)$ sub-pixels c (blue).

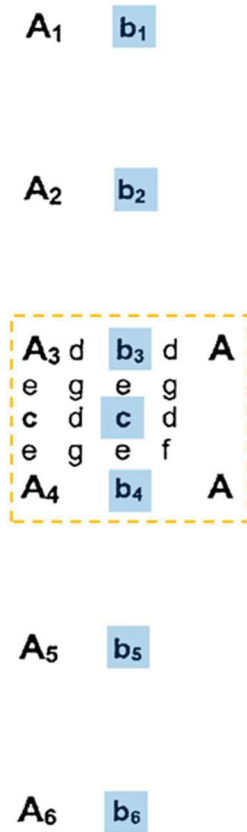


Figure 13(c)

EX-1001, Figure 13(c), 11:46-11:63 (“ $c=(b_1-5b_2+20b_3+20b_4-5b_5+b_6)$ ”); Ramchandran, ¶172. The ’273 patent further describes interpolating (2/4,2/4) sub-pixel c using a weighted sum of:

- (1) only one sub-pixel having co-ordinates with K equal to an even value (“2”) and L equal to zero (the (2/4,0/4) sub-pixel b_3) and a second (2/4,4/4) sub-pixel b_4 having co-ordinates with K equal to an even value (“2”) and L equal to four (and not zero), and

- (2) corresponding sub-pixels b_1 and b_2 and sub-pixels b_5 and b_6 in adjacent rectangular bounded regions.

EX-1001, Figures 14(a)-14(c), 45:23-43; Ramchandran, ¶171 n.17.

As shown below, Fandrianto interpolates highlighted $(2/4,2/4)$ sub-pixel $C_{9,10}$ either by (1) averaging circled $(2/4,0/4)$ sub-pixel $A_{9,10}$ and $(2/4,4/4)$ sub-pixel $A_{10,10}$ (vertical interpolation; left) or (2) averaging circled $(0/4,2/4)$ sub-pixel $B_{9,10}$ and $(4/4,2/4)$ sub-pixel $B_{9,11}$ (horizontal interpolation; right).

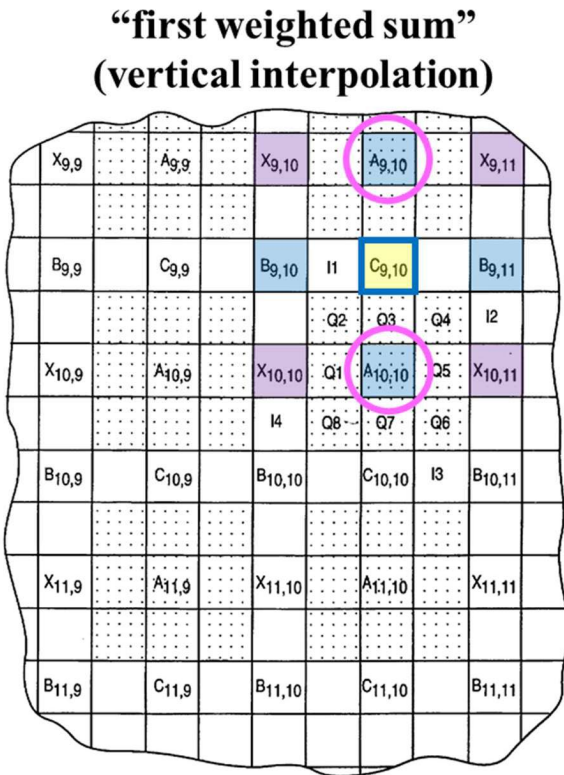


FIG. 5

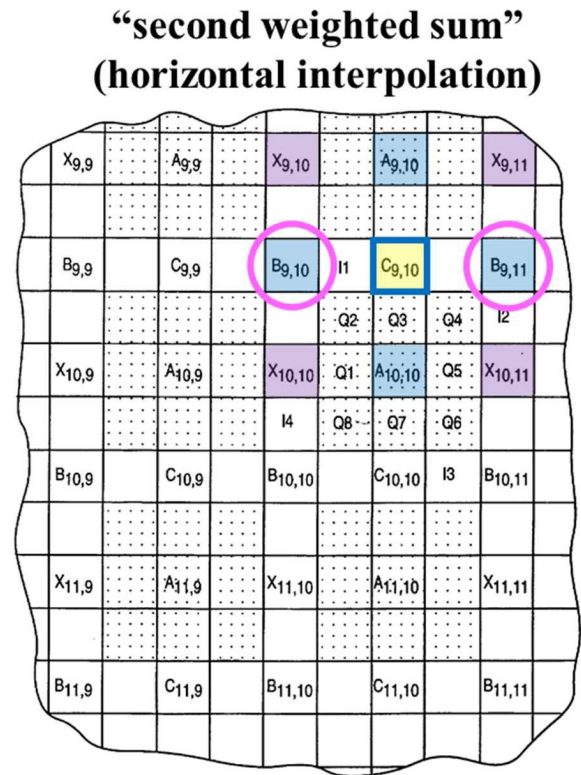
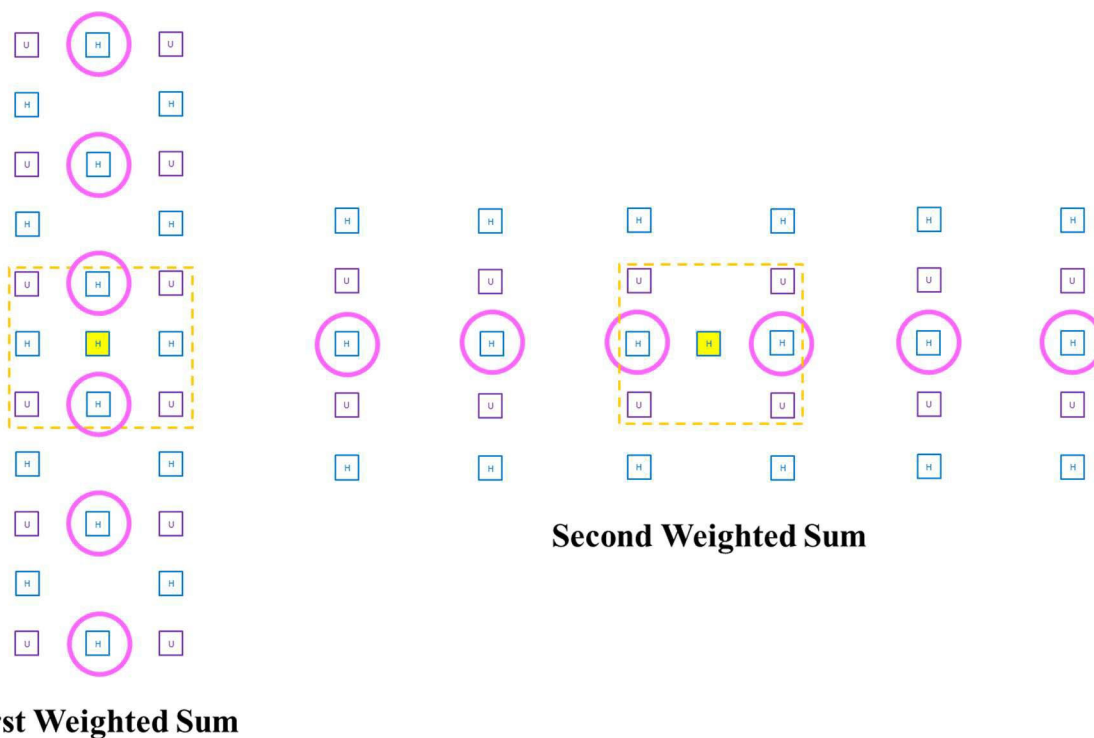


FIG. 5

Fandrianto, 5:26-32, 6:4-12, 6:49-6:54, 6:18-48, 8:36-10:50. Accordingly, Fandrianto interpolates the highlighted $(2/4,2/4)$ sub-pixel using either the “first weighted sum” of the values of sub-pixels having co-ordinates with K equal to an

even value 2 and L equal to zero (“A block”) or another weighted sum (“second weighted sum”) of the values of sub-pixels having co-ordinates with K equal to zero and L equal to an even value (“B block”), with respective weights in each weighted sum equal to “1.” *Id.*; Ramchandran, ¶173.

As discussed *supra* §VIII.A.1, it would have been obvious for a POSITA to modify TML6 using Fandrianto such that *oneforthpix()* interpolates (2/4,2/4) sub-pixels using a predetermined choice of either (1) the first weighted sum of TML6 (summing six sub-pixels having co-ordinates with K equal to an even value (2) and L equal to zero) or (2) the second weighted sum of TML6 as modified by Fandrianto (also summing six sub-pixels, but horizontally like Fandrianto with sub-pixels having co-ordinates with K equal to zero and L equal to an even value (2)), as shown below.



Ramchandran, ¶174.

Fandrianto explicitly teaches a finite number of interpolation techniques for the (2/4,2/4) sub-pixels, including vertical interpolation using a first weighted sum (as also used in TML6), and horizontal interpolation using a second weighted sum. Fandrianto, 6:4-12, 6:49-7:2; Ramchandran, ¶174. Thus, it would have been obvious to try these finite options, including calculating both first and second weighted sums and using a predetermined choice between the two to interpolate the (2/4,2/4) sub-pixels in TML6 using six sub-pixels, consistent with TML6. Ramchandran, ¶174. Further, revising TML6 code would have been straightforward and yielded the predictable result of interpolating (2/4,2/4) sub-pixels. Ramchandran, ¶174. In fact, the '273 patent recognizes that interpolating (2/4,2/4) sub-pixels using either a

vertical weighted sum of sub-pixels or a horizontal weighted sum of sub-pixels “leads to the same result.” EX-1001, Figure 14(a), 42:18-52, 45:25-43; Ramchandran, ¶174.

Accordingly, the TML6-Fandrianto combination teaches a predetermined choice between the weighted sums of values, as claimed. Ramchandran, ¶¶174-175.

b. Claim 4

The TML6-Fandrianto combination teaches Claim 4’s additional features. As discussed for 1[a] and illustrated below, Fandrianto interpolates the highlighted (1/4,3/4) sub-pixel Q2 having co-ordinates ($K/2^N, L/2^N$; $N=2$) with odd values of K (**K=1**) and L (**L=3**) according to a predetermined choice of two weighted averages.



Fandrianto, Figure 5, 6:18-7:2, 8:36-10:58;¹⁵ Ramchandran, ¶176.

When the chosen weighted average involves the pair of diagonally-opposed (0/4,2/4) sub-pixel B and (2/4,4/4) sub-pixel A (blue; circled above) having co-ordinates ($K/2^N, L/2^N$; $N=2$) with even values of both K (**$K=0$ or 2**) and L (**$L=2$ or**

¹⁵ As explained *supra* §VII.B, A block 72 and B block 74 contain “A” and “B” data, respectively, because Fandrianto does not require a specific order for deriving pixels Q1-Q8. Ramchandran, ¶¶156-158 (citing Fandrianto, 7:26-8:20, 10:51-58, Figures 3B, 5).

4), the magnitudes of the first and second weights (“1”) in the weighted average of the sub-pixels A and B (blue) are inversely proportional to the respective straight-line diagonal distances (orange arrows) of the circled sub-pixels A and B (blue) (having co-ordinates with even values of both K and L) to the interpolated highlighted sub-pixel Q2 (having co-ordinates with odd values of both K and L).

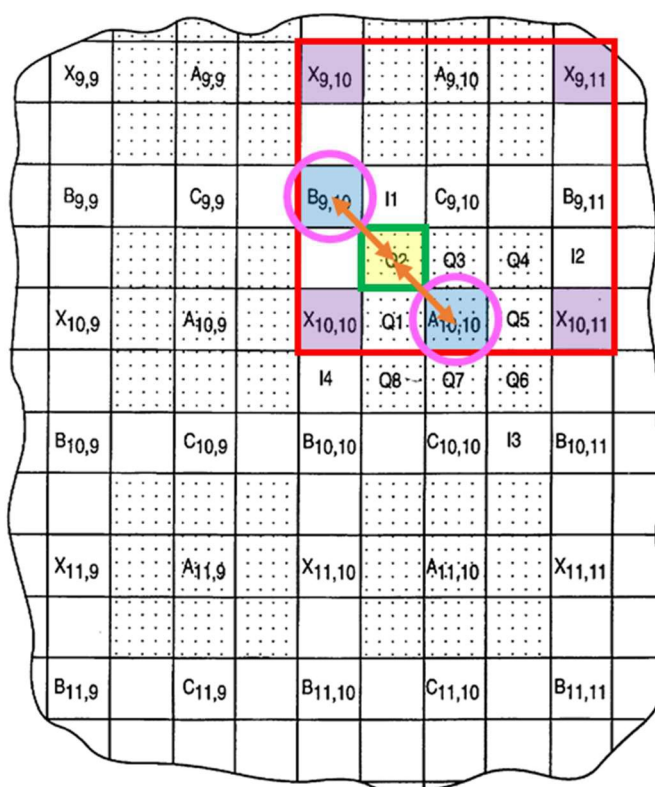


FIG. 5

Fandrianto, Figure 5; Ramchandran, ¶177. The sub-pixels A and B (blue) are equidistant from the sub-pixel Q2 (green), and thus, the magnitudes of their weights (“1”) are inversely proportional to their distances (orange arrows) from the sub-pixel Q2 (green). Ramchandran, ¶177.

If Claim 4 requires affirmatively “causing the apparatus to select” the relative magnitudes of the first and second weights of the weighted average to be inversely proportional to the respective distances of sub-pixels A and B to sub-pixel Q2, that would have been obvious. The closer A and B are to Q2, the more likely their values will be similar to Q2. Thus, the closer sub-pixels should have larger weights. Conversely, the farther A and B are from Q2, the less likely their values will be similar. Accordingly, the farther sub-pixels should have smaller weights. Thus, a POSITA would have known to select the relative magnitudes of the weights of A and B to be inversely proportional to their respective distances from Q2. Ramchandran, ¶¶178-179.

Accordingly, the TML6-Fandrianto combination interpolates values of TML6’s $(1/4, 1/4)$, $(1/4, 3/4)$, $(3/4, 1/4)$, and $(3/4, 3/4)$ sub-pixels having co-ordinates $(K/2^N, L/2^N; N=2)$ with odd values of both K (**K=1 or 3**) and L (**L=1 or 3**), as claimed. Ramchandran, ¶180.

c. Claim 5

The TML6-Fandrianto combination teaches Claim 5’s additional features. As discussed for Claim 4 and shown below, the first and second weights have equal values (“1”) when sub-pixels A and B (blue)—having co-ordinates with even values of both K and L—are equidistant from the highlighted sub-pixel Q2—having co-ordinates with odd values of both K and L—being interpolated.

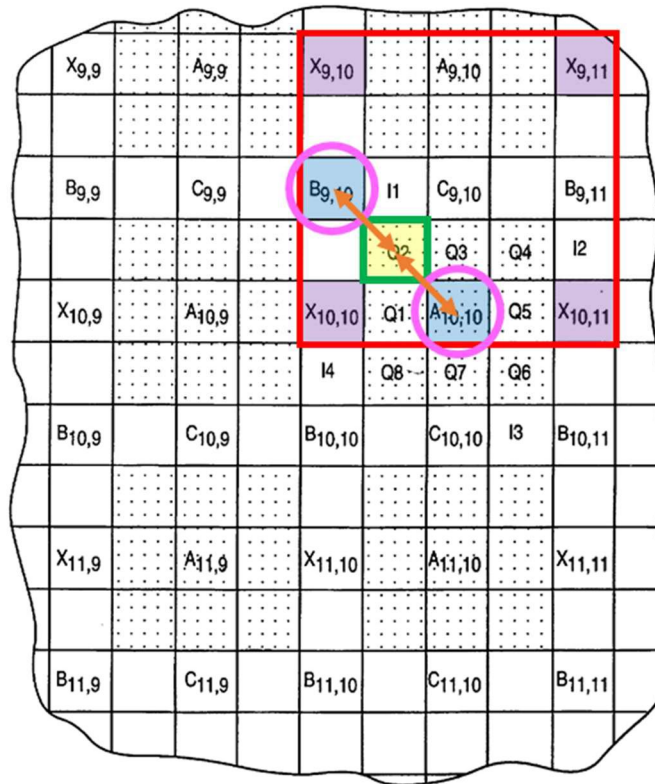


FIG. 5

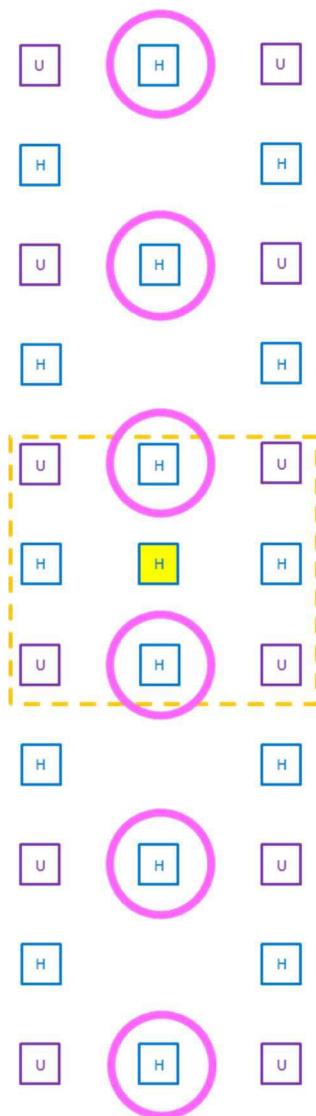
Fandrianto, Figure 5; Ramchandran, ¶181.

d. Claim 6

The TML6-Fandrianto combination teaches Claim 6's additional features. As discussed for 1[c], values for sub-pixels having co-ordinates with even values of K and L—e.g., (2/4,2/4) sub-pixels—along with sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions are interpolated using a predetermined choice of a first weighted sum of certain sub-pixel values and a second weighted sum of other sub-pixel values. Claim 6 repeats verbatim much of 1[c] and further states that the sub-pixel values are interpolated using the first

weighted sum “when the value of a sub-pixel having co-ordinates with K equal to an even value and L equal to an odd value is also required.” Ramchandran, ¶¶182-183.

As discussed for 1[c] and illustrated below, TML6 interpolates highlighted (2/4,2/4) sub-pixels using the first weighted sum of the sub-pixel values (circled).



Ramchandran, ¶183. Furthermore, Fandrianto interpolates highlighted (2/4,1/4) sub-pixel Q7 having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to an even value

($K=2$) and L equal to an odd value ($L=1$) and interpolates corresponding Q7 sub-pixels in other rectangular bounded regions, when required for motion estimation. Sub-pixel Q7 is vertically interpolated using sub-pixel $A_{10,10}$ and $(2/4,2/4)$ sub-pixel $C_{10,10}$.

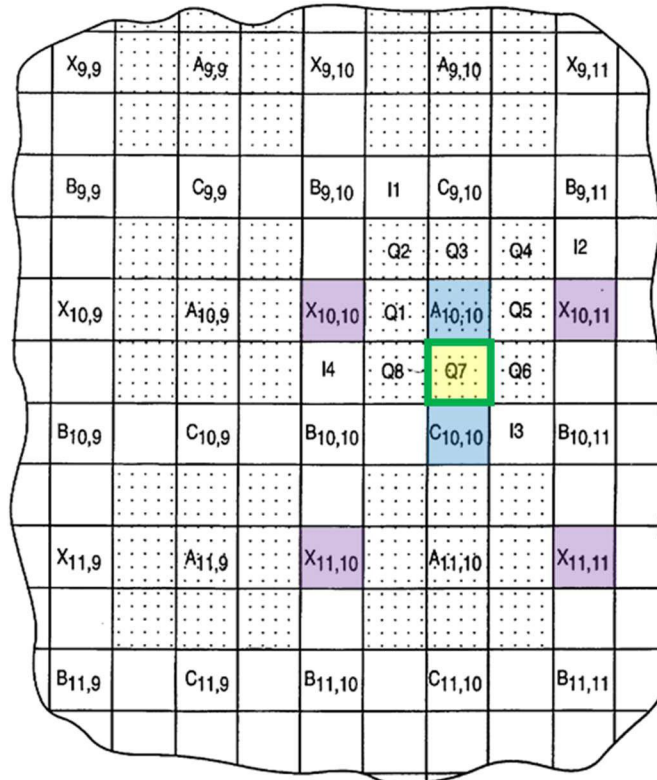
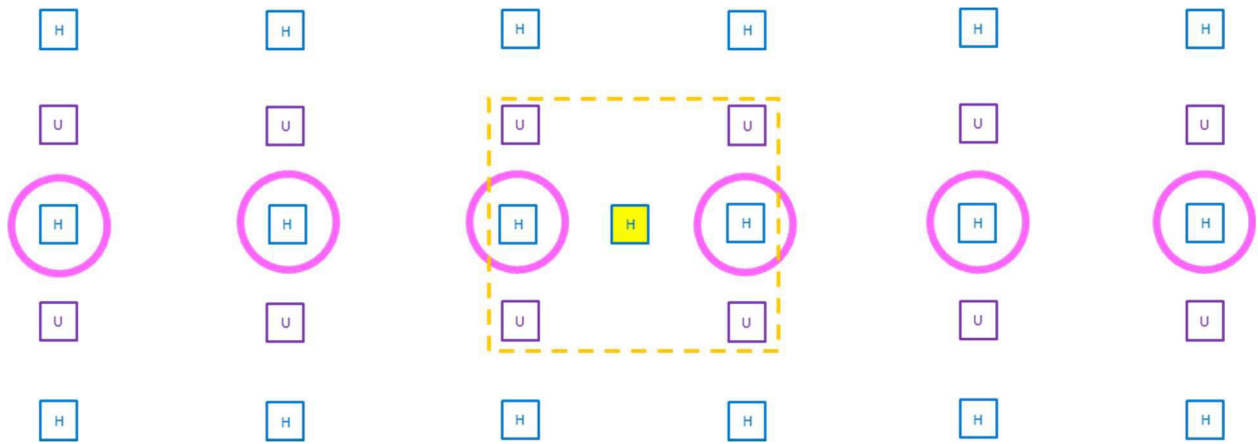


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-35, 10:23-33; *see id.*, 8:36-57 (“Q1”), 9:33-43 (“Q3”). Accordingly, sub-pixel $C_{10,10}$ is also interpolated when interpolating the sub-pixel Q7 is required. Fandrianto, 5:26-7:19, 8:29-35, 10:23-33 (“ $Q7_{r,c}=(C_{r,c}+A_{r,c})/2$ ”). The TML6-Fandrianto combination thus interpolates $(2/4,2/4)$ sub-pixels using the first weighted sum (shown below) when interpolating the $(2/4,1/4)$ sub-pixel is also required.

values are interpolated using the second weighted sum “when the value of a sub-pixel having co-ordinates with K equal to an odd value and L equal to an even value is also required.” Ramchandran, ¶¶185-186.

As explained for 1[c] and shown below, the TML6-Fandrianto combination also interpolates these highlighted (2/4,2/4) sub-pixels using the second weighted sum of the circled sub-pixels.



Ramchandran, ¶187.

Fandrianto also interpolates highlighted (1/4,2/4) sub-pixel I1 having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to an odd value (**K=1**) and L equal to an even value (**L=2**), and corresponding Q7 sub-pixels in other rectangular bounded regions, when required for motion estimation.

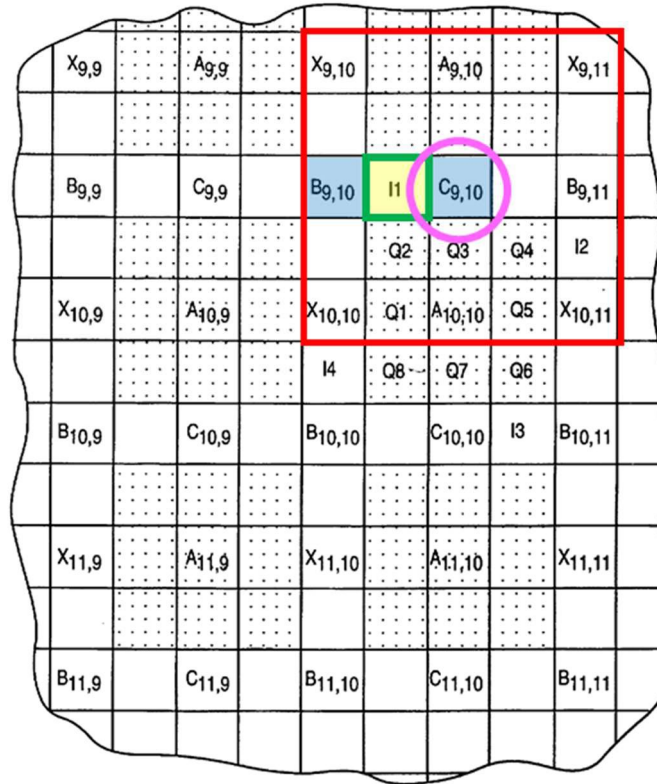
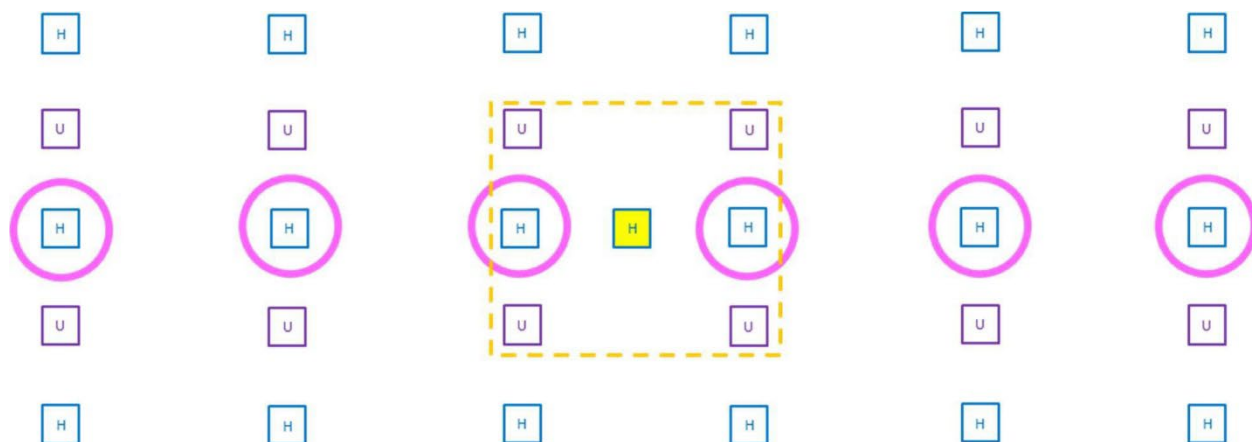


FIG. 5

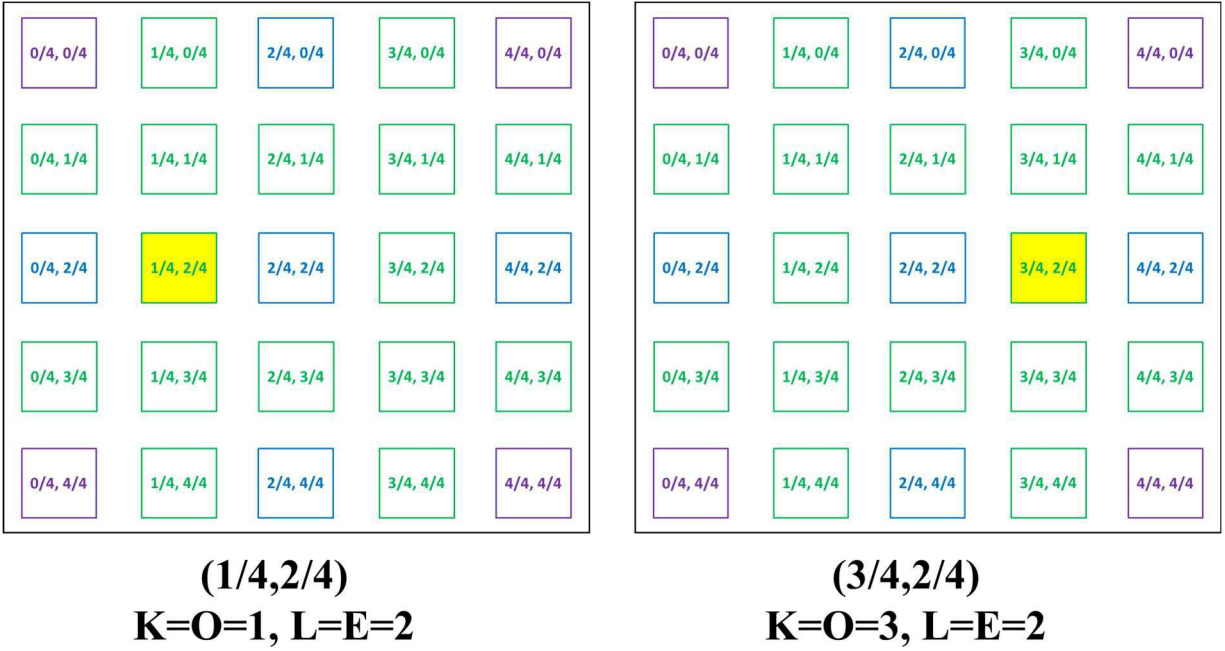
Fandrianto, Figure 5, 5:26-32, 8:29-35, 8:58-9:32 (“ $I_{1,r,c} = (B_{r-1,c} + C_{r-1,c})/2$ ”) 10:6-22 (“I3”). As shown above, interpolating the highlighted I1 sub-pixel uses (2/4,2/4) sub-pixel C_{9,10} (blue), and thus, sub-pixel C_{9,10} would be interpolated when interpolating the sub-pixel I1 is required. Fandrianto, Figure 5, 5:26-7:19, 8:29-35, 8:58-9:32 (“ $I_{1,r,c} = (B_{r-1,c} + C_{r-1,c})/2$ ”); *see id.*, 10:6-22. Therefore, the TML6-Fandrianto combination’s sub-pixel interpolation technique interpolates (2/4,2/4) sub-pixels using the second weighted sum (shown below) when the (1/4,2/4) sub-pixel is also required.



Id.; Ramchandran, ¶188.

f. Claim 8

The TML6-Fandrianto combination teaches Claim 8's additional features. As shown below, the combination interpolates sub-pixels—e.g., sub-pixels (1/4,2/4) and (3/4,2/4), highlighted—having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to an odd value O ($K=O=1$ or 3), and L equal to an even value E ($L=E=2$), which is not zero or 2^N ($2^2=4$).



Ramchandran, ¶189.

As shown below (left), Fandrianto interpolates the highlighted (1/4, 2/4) sub-pixel I1 by averaging first (0/4, 2/4) sub-pixel B_{9,10} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to the even value O-1 (**K=1-1=0**) and L equal to E (**L=E=2**) and a second (2/4, 2/4) sub-pixel C_{9,10} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to the even value O+1 (**K=1+1=2**) and L equal to E (**L=E=2**). Similarly, as shown below (right), Fandrianto interpolates the highlighted (3/4, 2/4) sub-pixel I3 by averaging first (2/4, 2/4) sub-pixel C_{10,10} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to the even value O-1 (**K=3-1=2**) and L equal to E (**L=E=2**) and a second (4/4, 2/4) sub-pixel B_{10,11} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to the even value O+1 (**K=3+1=4**) and L equal to E (**L=E=2**).

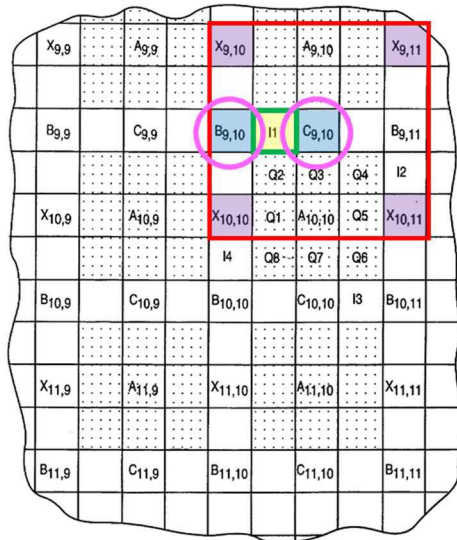


FIG. 5

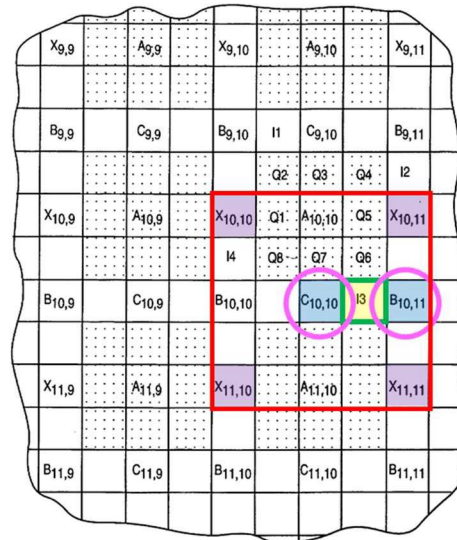


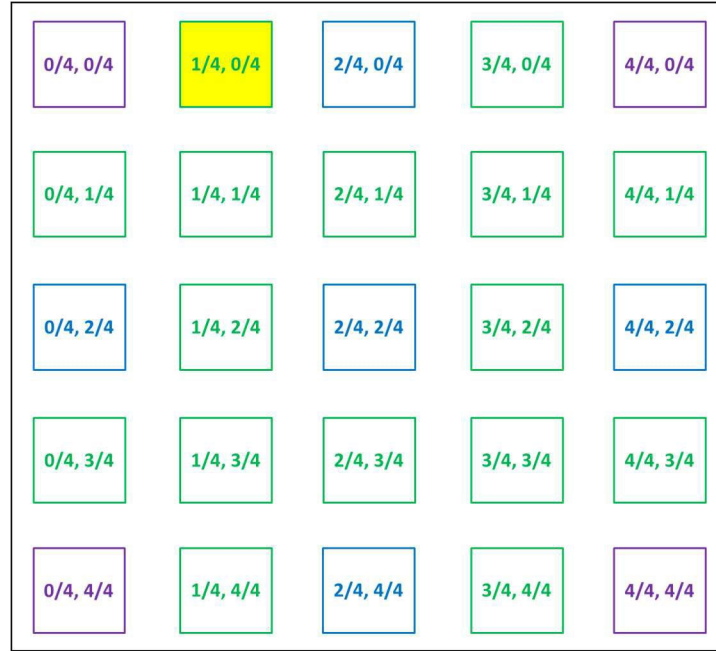
FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 8:58-9:32 (“ $I_{1,r,c}=(B_{r-1,c}+C_{r-1,c})/2$ ”), 10:6-22 ($I_{3,r,c}=(B_{r,c+1}+C_{r,c})/2$); Ramchandran, ¶190.

As discussed *supra* §VIII.A.1, implementing Fandrianto’s technique to interpolate the claimed sub-pixels using values in TML6’s *mref* was obvious. Ramchandran, ¶191.

g. Claim 9

The TML6-Fandrianto combination teaches Claim 9’s additional features. As shown below, the combination interpolates the highlighted (1/4,0/4) sub-pixel having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 1 and L equal to zero.



(1/4,0/4)
K=1, L=0

Ramchandran, ¶192.

Furthermore, as shown below, Fandrianto interpolates highlighted (1/4,0/4) sub-pixel Q1 by averaging (0/4,0/4) pixel $X_{10,10}$ (purple; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to zero and L equal to zero and (2/4,0/4) sub-pixel $A_{10,10}$ (blue; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2 and L equal to zero.

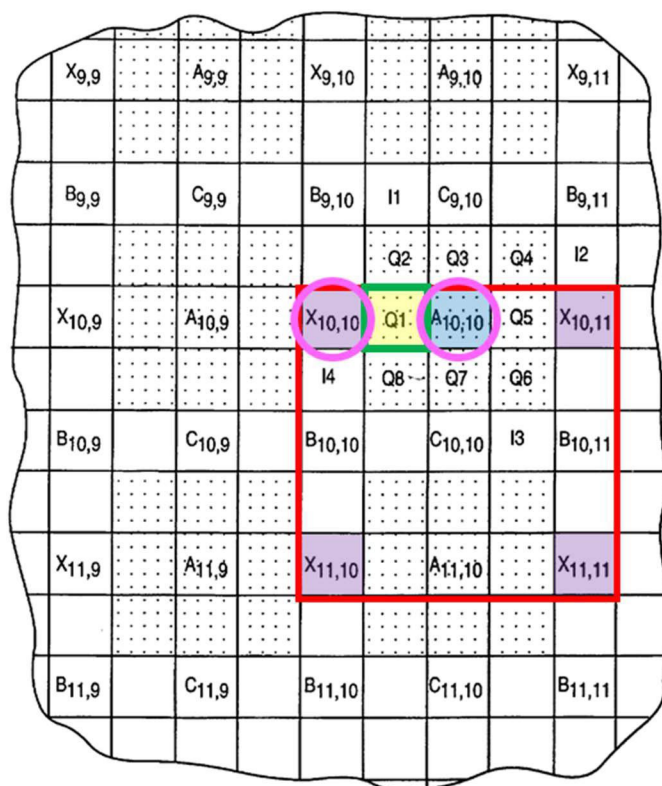
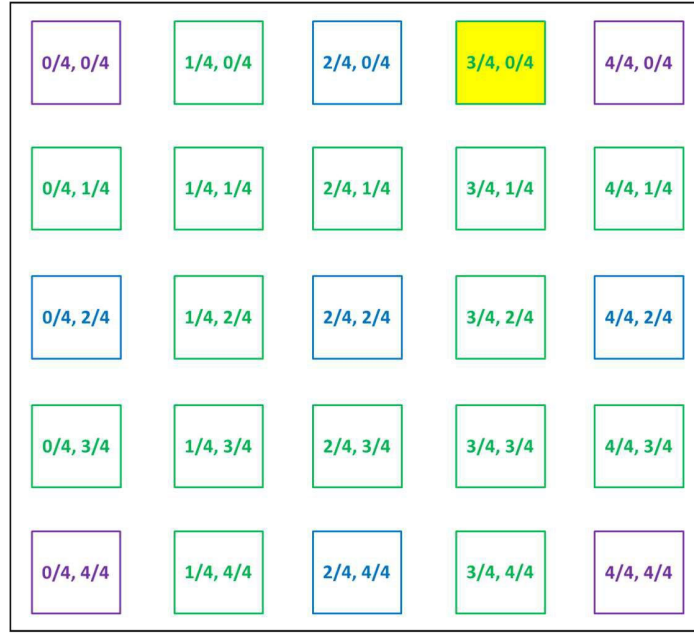


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57 (“ $Q1_{r,c} = (X_{r,c} + A_{r,c})/2$ ”); Ramchandran, ¶¶193-194.

h. Claim 10

The TML6-Fandrianto combination teaches Claim 10’s additional features. As shown below, the combination interpolates highlighted (3/4,0/4) sub-pixel having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N-1 ($K=2^2-1=3$) and L equal to zero.



(3/4,0/4)
K=3, L=0

Ramchandran, ¶195.

Furthermore, as shown below, Fandrianto interpolates highlighted (3/4,0/4) sub-pixel Q5 by averaging (4/4,0/4) pixel $X_{10,11}$ (purple; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N ($K=2^2=4$) and L equal to zero, and (2/4,0/4) sub-pixel $A_{10,10}$ (blue; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N-2 ($K=2^2-2=2$) and L equal to zero.

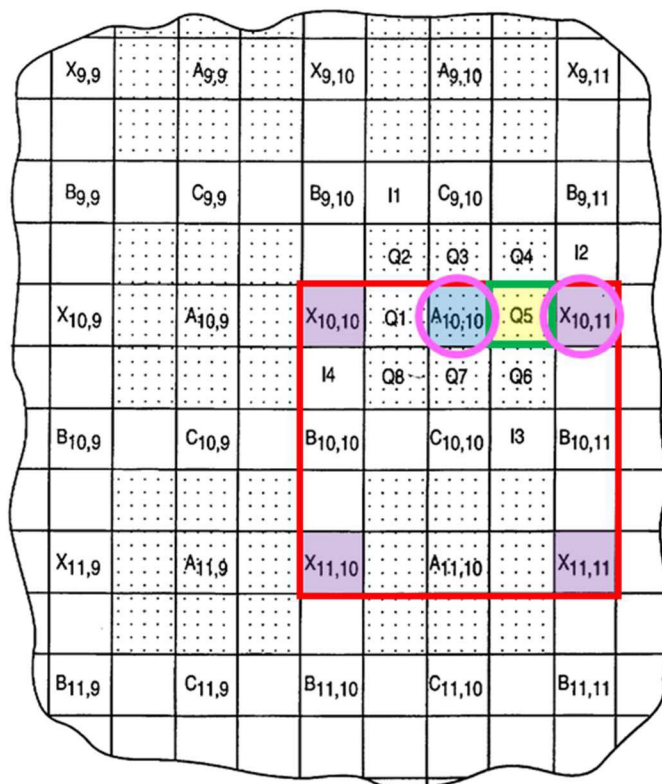
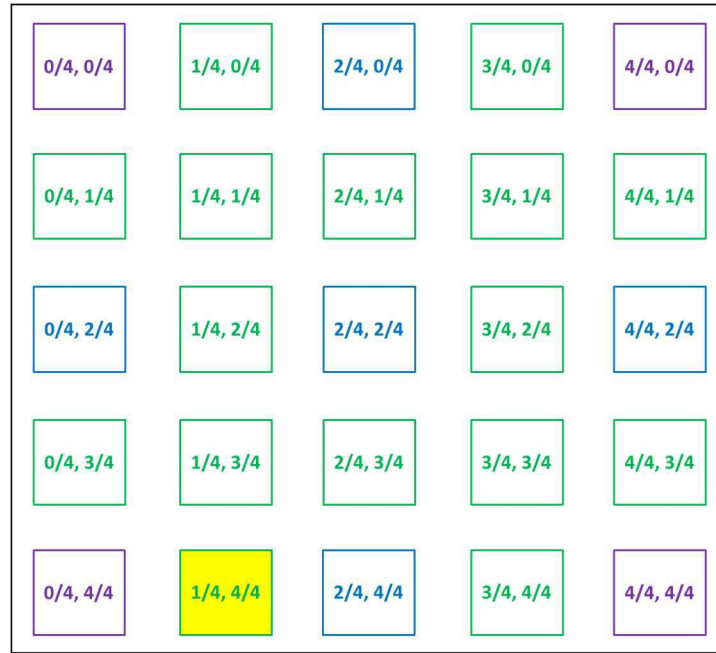


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 9:63-10:5 (“ $Q5_{r,c}=(X_{r,c+1}+A_{r,c})/2$ ”);
Ramchandran, ¶¶196-197.

i. Claim 11

The TML6-Fandrianto combination teaches Claim 11’s additional features.
As shown below, the combination interpolates highlighted (1/4,4/4) sub-pixel having
co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 1 and L equal to 2^N ($L=2^2=4$).



(1/4,4/4)
K=1, L=4

Ramchandran, ¶198.

Furthermore, as shown below, Fandrianto interpolates highlighted (1/4,4/4) sub-pixel Q1 by averaging (0/4,4/4) pixel $X_{10,10}$ (purple; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to zero and L equal to 2^N ($L=2^2=4$), and (2/4,4/4) sub-pixel $A_{10,10}$ (blue; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2 and L equal to 2^N ($L=2^2=4$).

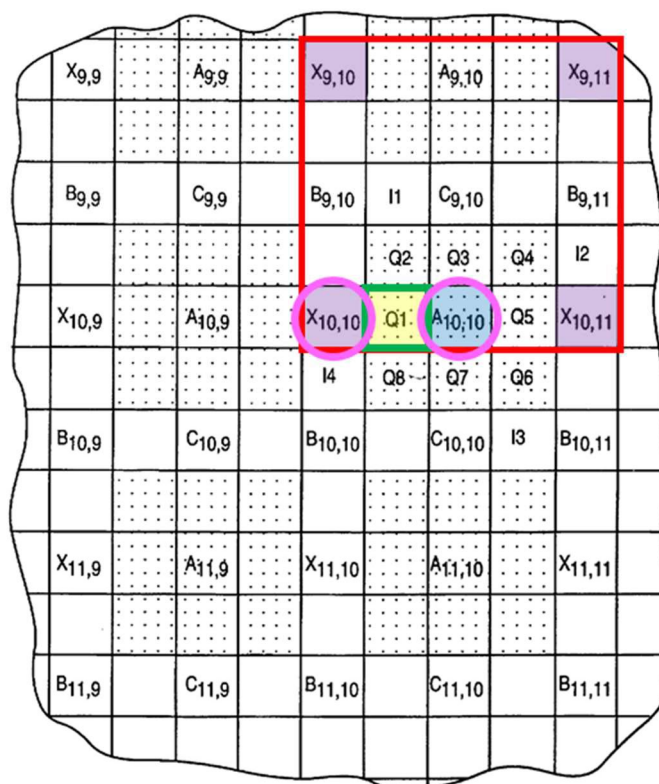
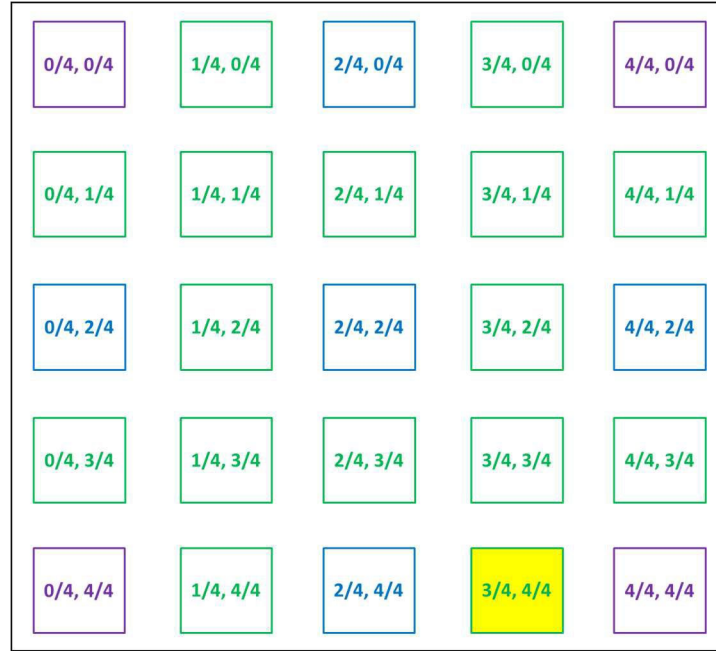


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57 (“ $Q_{1,r,c} = (X_{r,c} + A_{r,c})/2$ ”); Ramchandran, ¶¶199-200.

j. Claim 12

The TML6-Fandrianto combination teaches Claim 12's additional features. As shown below, the combination interpolates highlighted (3/4,4/4) sub-pixel having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to 2^N-1 (**K=2²-1=3**) and L equal to 2^N (**L=2²=4**).



$\frac{3}{4}, \frac{4}{4}$
K=1, L=4

Ramchandran, ¶201.

Furthermore, as shown below, Fandrianto interpolates highlighted $(\frac{3}{4}, \frac{4}{4})$ sub-pixel Q5 by averaging $(\frac{4}{4}, \frac{4}{4})$ pixel $X_{10,11}$ (purple; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N ($K=2^2=4$) and L equal to 2^N ($L=2^2=4$), and $(\frac{2}{4}, \frac{4}{4})$ sub-pixel $A_{10,10}$ (blue; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N-2 ($K=2^2-2=4$) and L equal to 2^N ($L=2^2=4$).

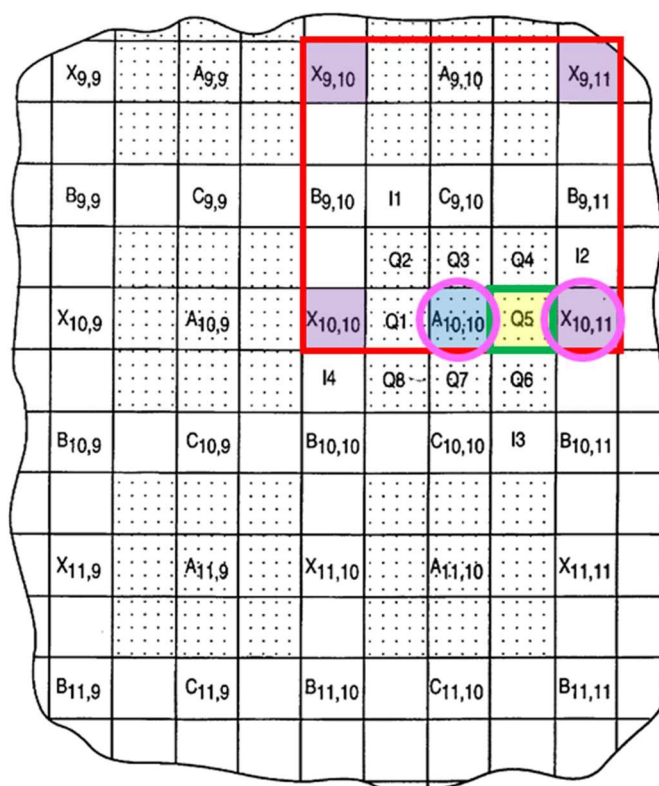
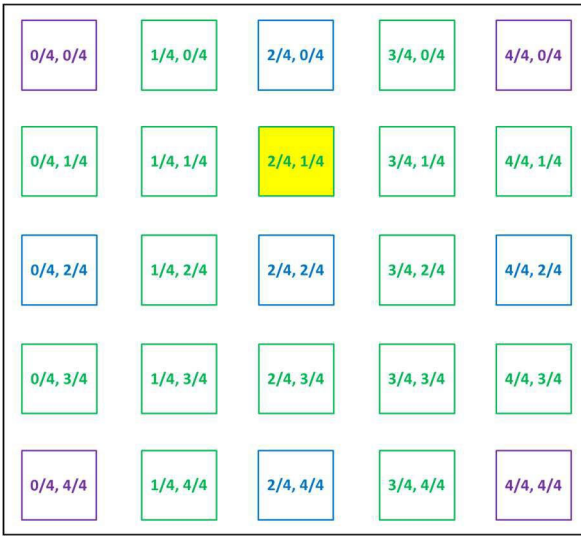


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 9:63-10:5 (“ $Q5_{r,c}=(X_{r,c+1}+A_{r,c})/2$ ”);
Ramchandran, ¶¶202-203.

k. Claim 13

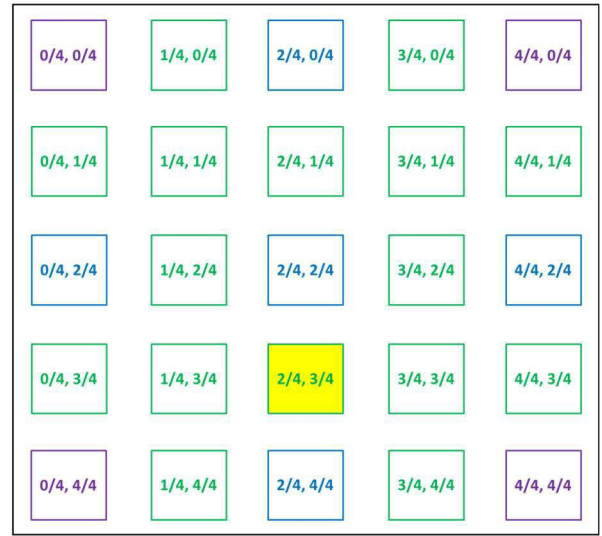
The TML6-Fandrianto combination teaches Claim 13’s additional features. As shown below and discussed for 1[a], the combination interpolates values for the highlighted (2/4,1/4) and (2/4,3/4) sub-pixels having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to an even value E ($K=E=2$), not including zero or 2^N ($2^2=4$) and L equal to an odd value O ($L=O=1$ or 3).



(2/4, 1/4)

K=2, L=1

K=E=2, L=O=1



(2/4, 3/4)

K=2, L=3

K=E=2, L=O=3

Ramchandran, ¶204.

Furthermore, as shown below (left), Fandrianto interpolates highlighted (2/4, 1/4) sub-pixel Q7 by averaging first (2/4, 0/4) sub-pixel A_{10,10} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to E (**K=E=2**) and L equal to the even value O-1 (**L=1-1=0**) and second (2/4, 2/4) sub-pixel C_{10,10} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to E (**K=E=2**) and L equal to the even value O+1 (**L=1+1=2**). As shown below (right), Fandrianto interpolates highlighted (2/4, 3/4) sub-pixel Q3 by averaging first (2/4, 2/4) sub-pixel C_{9,10} (blue; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to E (**K=E=2**) and L equal to the even value O-1 (**L=3-1=2**) and second (2/4, 4/4) sub-pixel A_{10,10} (blue;

circled) having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to E ($K=E=2$) and L equal to the even value $O+1$ ($L=3+1=4$).

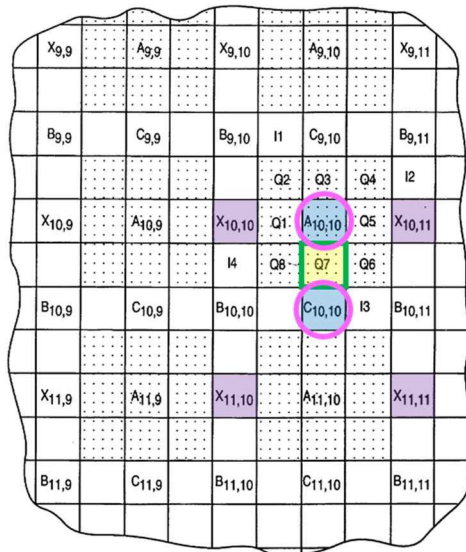


FIG. 5

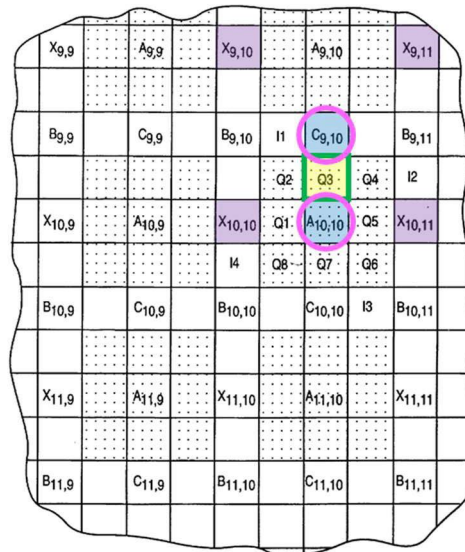
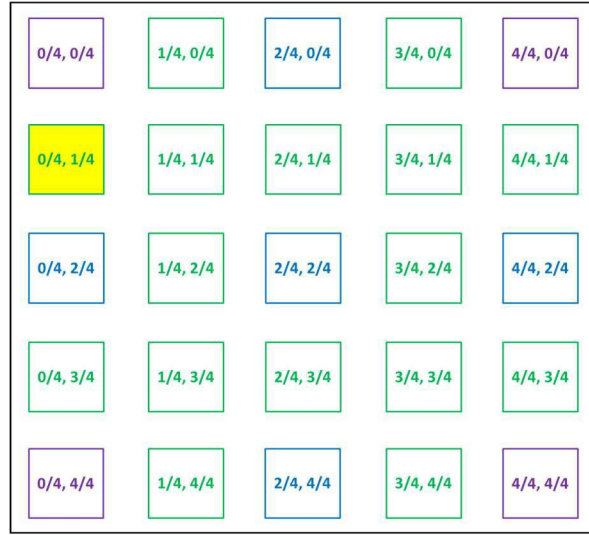


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 10:23-33, 9:32-44; Ramchandran, ¶¶205-206.

l. Claim 14

The TML6-Fandrianto combination teaches Claim 14's additional features. As shown below, the combination interpolates values for the highlighted (0/4,1/4) sub-pixels having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to zero and L equal to 1.



(0/4,1/4)
K=0, L=1

Ramchandran, ¶207.

As shown below, Fandrianto interpolates highlighted (0/4,1/4) sub-pixel I₄ by averaging (0/4,0/4) pixel X_{10,10} (purple; circled) having co-ordinates (K/2^N,L/2^N; N=2) with both K and L equal to zero and (0/4,2/4) sub-pixel B_{10,10} (blue; circled) having co-ordinates (K/2^N,L/2^N; N=2) with K equal to zero and L equal to 2.

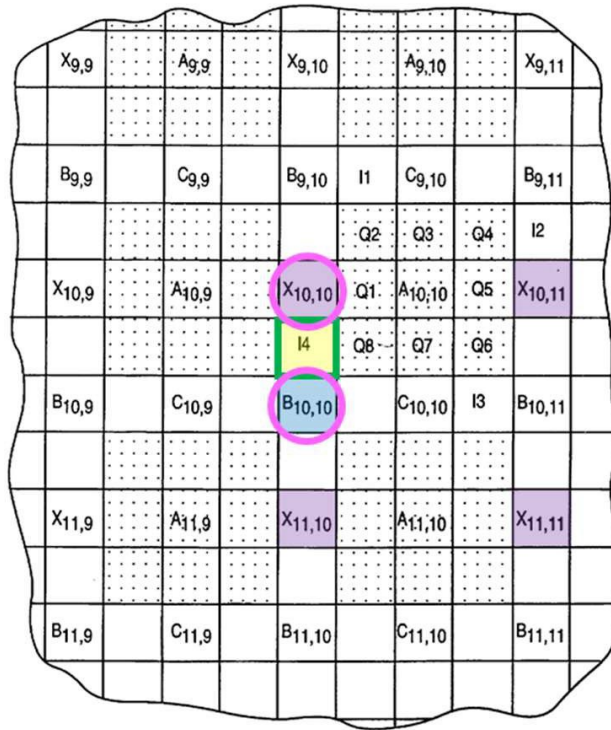
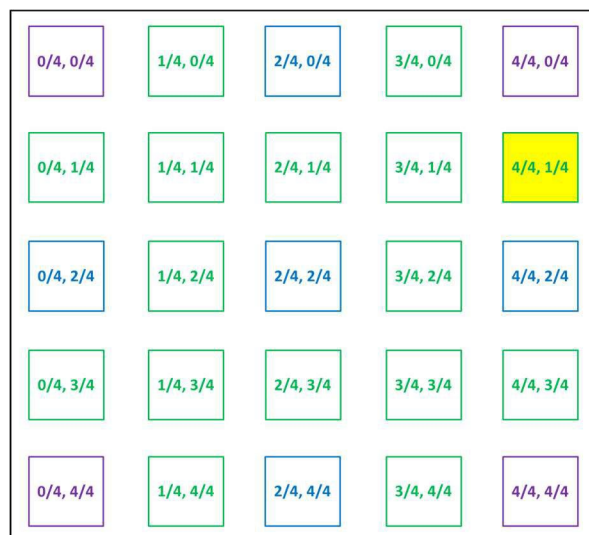


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 10:34-10:50 (“ $I4_{r,c}=(B_{r,c}+X_{r,c})/2$ ”);
Ramchandran, ¶¶208-209.

m. Claim 15

The TML6-Fandrianto combination teaches Claim 15’s additional features.
As shown below, the combination interpolates highlighted (4/4,1/4) sub-pixels
having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N ($K=2^2=4$) and L equal to
1.



(4/4,1/4)
K=4, L=1

Ramchandran, ¶210.

As shown below, Fandrianto interpolates highlighted (4/4,1/4) sub-pixel I4 by averaging (4/4,0/4) pixel $X_{10,10}$ (purple; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N (**K=2²=4**) and L equal to zero, and (4/4,2/4) sub-pixel $B_{10,10}$ (blue; circled) having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to 2^N (**K=2²=4**) and L equal to 2.

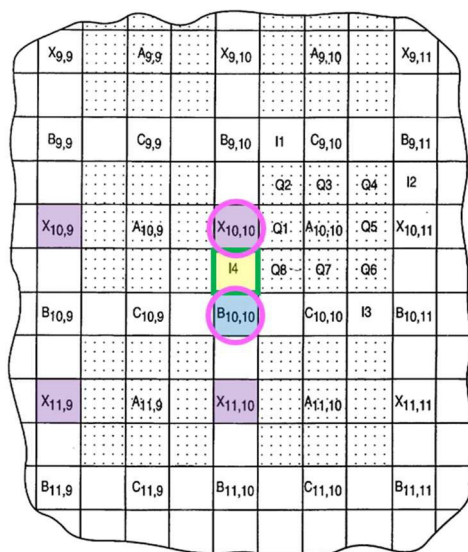
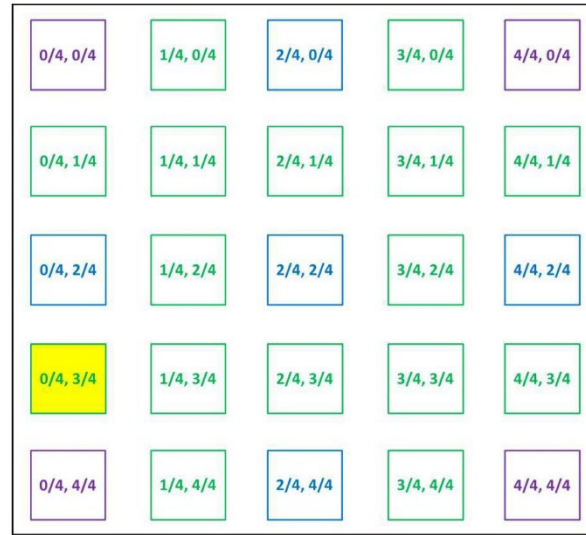


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 10:34-10:50; Ramchandran, ¶¶211-212.

n. Claim 16

The TML6-Fandrianto combination teaches Claim 16's additional features. As shown below and discussed for 1[a], the combination interpolates highlighted (0/4,3/4) sub-pixel having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to zero and L equal to 2^N-1 ($L=2^2-1=3$).



(0/4,3/4)
K=0, L=3

Ramchandran, ¶213.

As shown below,¹⁶ Fandrianto interpolates highlighted (0/4,3/4) sub-pixel I₂ by averaging (0/4,4/4) pixel X_{10,11} (purple; circled) having co-ordinates (K/2^N,L/2^N; N=2) with K equal to zero and L equal to 2^N (L=2²=4), and (0/4,2/4) sub-pixel B_{9,11} (blue; circled) having co-ordinates (K/2^N,L/2^N; N=2) with K equal to zero and L equal to 2^N-2 (L=2²-2=2).

¹⁶ The figure has been annotated to illustrate unit pixels X_{9,12} and X_{10,12}.

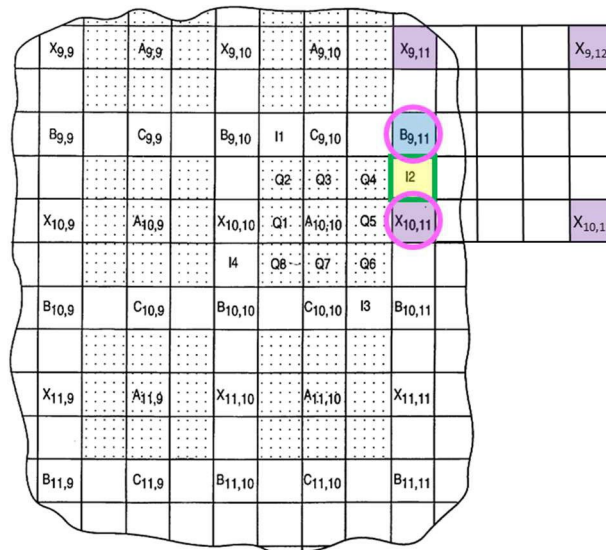
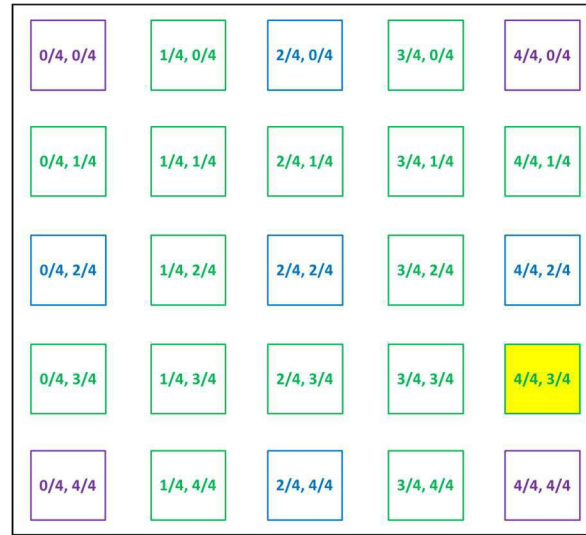


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 9:45-62; Ramchandran, ¶¶214-215.

o. Claim 17

The TML6-Fandrianto combination teaches Claim 17's additional features. As shown below, the combination interpolates highlighted (4/4,3/4) sub-pixel having co-ordinates ($K/2^N, L/2^N$; $N=2$) with K equal to 2^N ($K=2^2=4$) and L equal to 2^N-1 ($L=2^2-1=3$).



(4/4, 3/4)
K=4, L=3

Ramchandran, ¶216.

As shown below, Fandrianto interpolates highlighted (4/4, 3/4)¹⁷ sub-pixel I2, as defined by unit pixels (purple), by averaging (4/4, 4/4) pixel X_{10,11} (purple; circled) having co-ordinates (K/2^N, L/2^N; N=2) with K equal to 2^N (**K=2²=4**) and L equal to 2^N (**L=2²=4**) and (4/4, 2/4) sub-pixel B_{9,11} (blue; circled) having co-ordinates with K equal to 2^N (**K=2²=4**) and L equal to 2^N-2 (**L=2²-2=2**).

¹⁷ Under the claimed co-ordinate notation (1[pre][b], *supra*), sub-pixel 4/4, 3/4 is equivalent to sub-pixel 0/4, 3/4 (discussed for Claim 16, *supra*) because 4/4, 3/4 is defined by the rectangular bounded region about C_{9,11} and 0/4, 3/4 is defined by the rectangular bounded region about C_{9,10}.

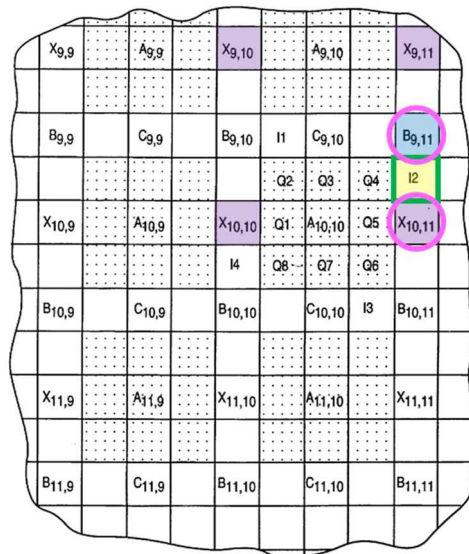


FIG. 5

Fandrianto, Figure 5, 5:26-32, 8:29-57, 9:45-62; Ramchandran, ¶¶217-218.

p. Claim 19

As discussed for 1[pre][b], TML6 teaches $N=2$. Ramchandran, ¶219.

q. Claim 20

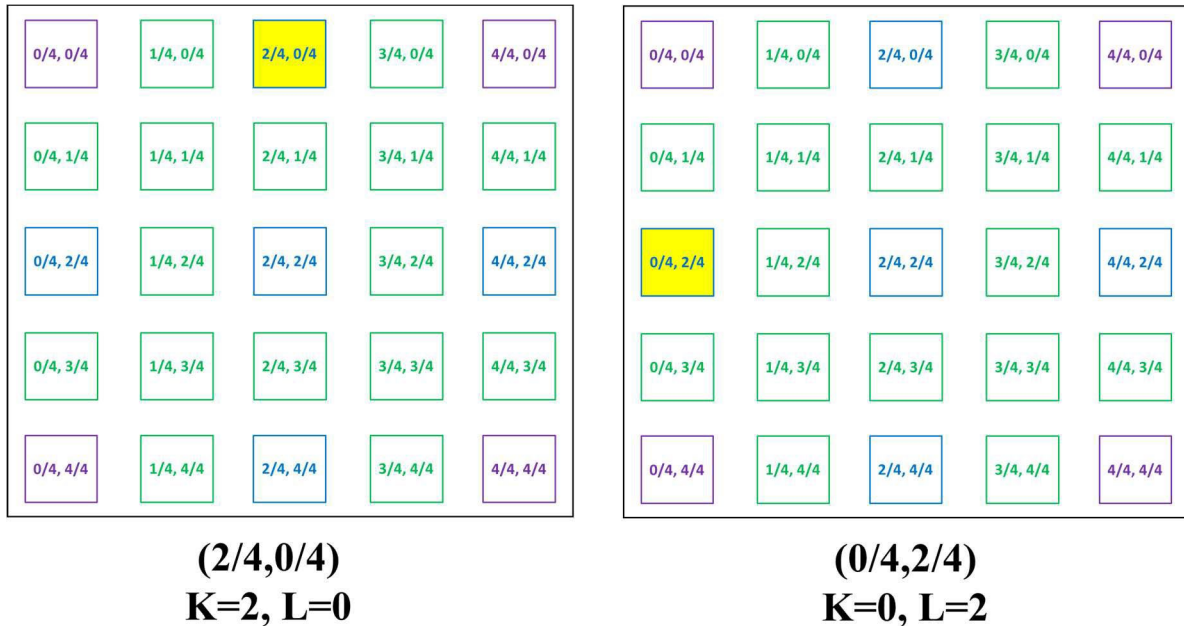
The TML6-Fandrianto combination teaches Claim 20's additional features. TML6 discloses that *imgY* and *mref* store 8-bit values representing integers in a dynamic range from 0 to 255 that represent the values that the pixels can take.¹⁸ TML6, 57-58 (lines 563-625), 131 (lines 131-132), 72-73 (lines 215-221 ("currently

¹⁸ The '273 patent confirms TML6 discloses a dynamic range of "a minimum value of 0 and a maximum value of 2^n-1 , where n is the number of bits reserved for a pixel value...typically 8." EX-1001, 9:54-64. The range 0 to 2^8-1 represents a dynamic range of 0 to 255.

only 8 bit/pel is supported’’)). For example, values representing unit pixels in *imgY* that are within the dynamic range are multiplied by 1024 when written to *img4Y_tmp*. TML6, 57 (line 571). These *img4Y_tmp* values are normalized using division by 1024 (“/1024”) to return them to the dynamic range of 0 to 255 and stored in *mref* as integer values within that range (“max(0,min(255,(int)”). TML6, 58 (line 588); Ramchandran, ¶¶220-221.

r. Claim 21

The TML6-Fandrianto combination teaches Claim 21’s additional features. As discussed for 1[b] and shown below, TML6 interpolates (2/4,0/4) and (0/4,2/4) sub-pixels using weighted sums of pixels (purple).



Ramchandran, ¶¶222-223.

Claim 21 further requires “interpolating [these] sub-pixel values...to produce a first intermediate value.” TML6 discloses two “alternative” first intermediate values for the (2/4,0/4) sub-pixel: (1) “*is*” and (2) “*is*32*”. Ramchandran, ¶224.

Regarding the first alternative, the highlighted (2/4,0/4) sub-pixels are interpolated using unit pixels (purple) stored in *imgY* to create first intermediate value “*is*”. Ramchandran, ¶225; TML6, 57 (lines 563-575). This first intermediate value “*is*” has a dynamic range (0 to 8,160) equal to the dynamic range of the pixel values (purple) from *imgY* used in the weighted sum (0 to 255) multiplied by a value (32) equal to the sum of the respective weights used in the weighted sum from *ONE_FOURTH_TAP* (20+20-5-5+1+1=32). *Id.*; TML6, 3 (lines 28-34); 1[b]-1[c], *supra*; Ramchandran, ¶225.

For the second alternative, the first intermediate value “*is*32*” representing (2/4,0/4) sub-pixels is written to *img4Y_tmp*. TML6, 57 (lines 563-575). This value “*is*32*” has a dynamic range (0 to 261,120) equal to the dynamic range (0 to 255) of pixels (purple) from *imgY* used in the weighted sum multiplied by a value (1024) equal to the sum of the respective weights used in the weighted sum, which are from *ONE_FOURTH_TAP* and multiplied by 32 (32*20+32*20-32*5-32*5+32*1+32*1=1024). *Id.*; TML6, 3 (lines 28-34); 1[b]-1[c], *supra*; Ramchandran, ¶226.

Similarly, as shown below, TML6 discloses that first intermediate value “*is*” for the (0/4,2/4) sub-pixel is calculated using unit pixels (purple) in *img4Y_tmp*.

0/4, 0/4	1/4, 0/4	2/4, 0/4	3/4, 0/4	4/4, 0/4
0/4, 1/4	1/4, 1/4	2/4, 1/4	3/4, 1/4	4/4, 1/4
0/4, 2/4	1/4, 2/4	2/4, 2/4	3/4, 2/4	4/4, 2/4
0/4, 3/4	1/4, 3/4	2/4, 3/4	3/4, 3/4	4/4, 3/4
0/4, 4/4	1/4, 4/4	2/4, 4/4	3/4, 4/4	4/4, 4/4

(0/4,2/4)
K=0, L=2

TML6, 57-58 (lines 576-592). This first intermediate value “*is*” has a dynamic range (0 to 261,120 (=0 to 255*1024)) equal to the dynamic range (0 to 255) of unit pixels (purple) used in the weighted sum multiplied by a value (1024) equal to the sum of the respective weights used in the weighted sum ($1024*20/32+1024*20/32-1024*5/32-1024*5/32+1024*1/32+1024*1/32=1024$). *Id.*; 1[b]-1[c], *supra*. These respective weights used in the weighted sum are calculated using values from *ONE_FOURTH_TAP*, multiplied by 1024 (from writing unit pixels (purple) to *img4Y_tmp*), and divided by 32 (from the calculation of “*is*”). *Id.*; TML6, 57 (lines 563-575), 3 (lines 28-34); Ramchandran, ¶227.

The '273 patent confirms TML6 interpolates (2/4,0/4) and (0/4,2/4) sub-pixels (blue) to produce a first intermediate value having a dynamic range equal to the predetermined dynamic range of the pixels (purple) used in the weighted sum multiplied by a value equal to the sum of the respective weights used in the weighted sum.

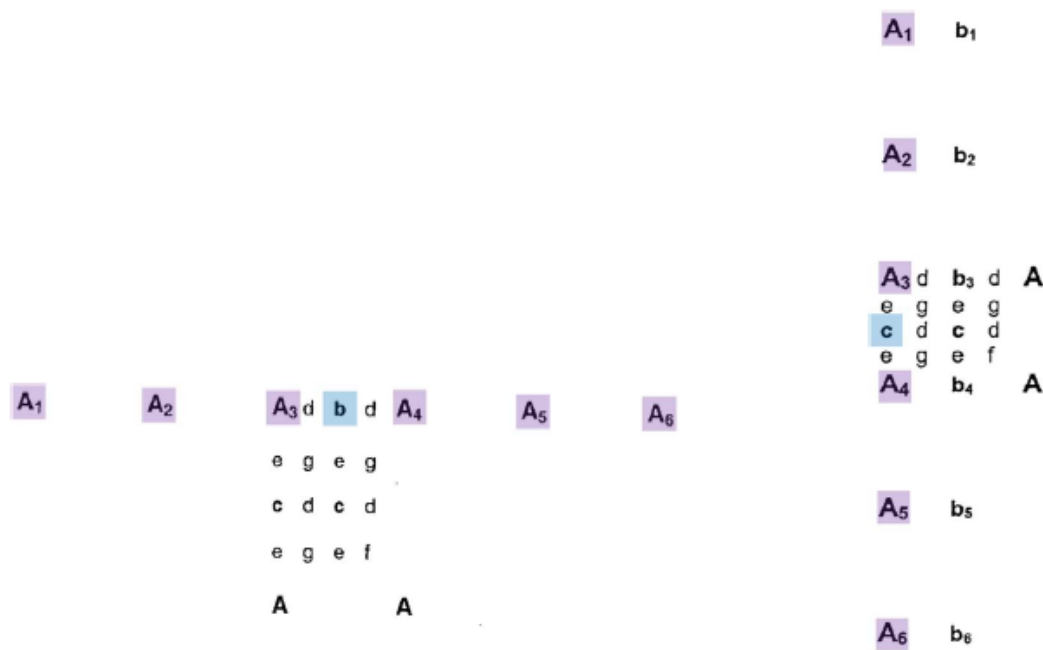


Figure13(b)

Figure 13(c)

EX-1001, Figures 13(b)-13(c), 11:35-63. Ramchandran, ¶228.

s. Claim 22

The TML6-Fandrianto combination teaches Claim 22's additional features. The dynamic range of the first intermediate value “*is*” for the (2/4,0/4) sub-pixel is (0 to 8,160). Claim 21, 1[b], *supra*. This range is representable by 13 bits (2¹³-

1=8,191 (required to represent 8160)). The dynamic range of the pixel values from *imgY* used to calculate the first intermediate value for the (2/4,0/4) sub-pixels is 0 to 255, which is defined by 8 bits ($2^8-1=255$). *Id.* The sum of the weights used in the weighted sum to calculate the first intermediate value for the (2/4,0/4) sub-pixels is 32, defined by 5 bits ($2^5=32$). *Id.* Thus, the dynamic range of the first intermediate value “*is*” for the (2/4,0/4) sub-pixel is defined by 13 bits (8+5 bits). *Id.*

The dynamic ranges of the alternate first intermediate value “*is*32*” for the (2/4,0/4) sub-pixel and the first intermediate value “*is*” for the (0/4,2/4) sub-pixel similarly meet the requirements of Claim 22 based on bits required to represent their respective weights. Ramchandran, ¶¶229-231.

t. Claim 23

The TML6-Fandrianto combination teaches Claim 23’s additional features. As discussed for Claim 21, TML6 discloses two “alternative” first intermediate values for the (2/4,0/4) sub-pixels: (1) “*is*” and (2) “*is*32*”. Ramchandran, ¶¶232-233.

The first intermediate value “*is*” for the (2/4,0/4) sub-pixel is truncated by dividing it by first scale factor **32**. The value “*is*” is initially multiplied by 32 (different than scale factor **32**) and written to *img4Y_tmp*. TML6, 58 (lines 552-592); Claim 21, *supra*. This value (“*is*”×32) in *img4Y_tmp* is divided by 1024

(= 32×32) to create a truncated first intermediate value that is written to *mref*. TML6, 58 (lines 563-592).

Accordingly, first intermediate value “*is*” is truncated by dividing it by a first scale factor **32**: ($\text{“is”} \times 32$)/(32×32). Moreover, the scale factor **32** equals the sum of the weights used in the weighted sum from *ONE_FOURTH_TAP* ($20+20-5-5+1+1=32$). TML6, 3 (lines 28-34); Claim 21, 1[b]-1[c], *supra*. Since “*is*” has a dynamic range of 0 to 8,160, dividing it by scale factor **32** produces a dynamic range of 0 to 255, equal to the dynamic range of the unit pixels ($8,160/32=255$). *Id.*; Ramchandran, ¶233.

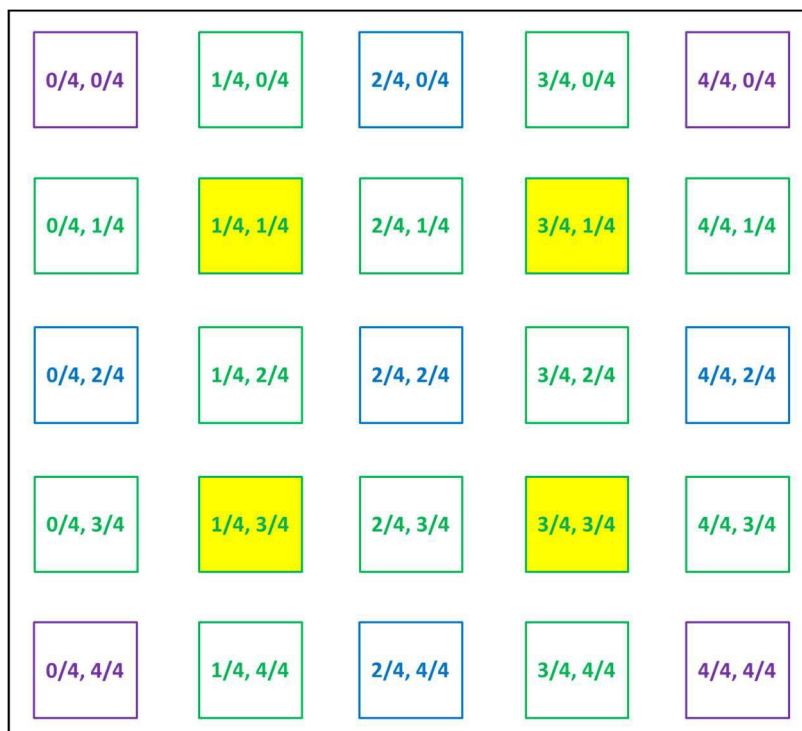
For the second alternative, the first intermediate value “*is*32*” for the (2/4,0/4) sub-pixel is truncated by dividing it by a first scale factor **1024**. As shown below, the highlighted value “*is*32*” is written to *img4Y_tmp*. TML6, 58 (lines 552-575); Claim 21, *supra*. When “*is*32*” is written from *img4Y_tmp* to *mref*, it is divided by first scale factor **1024**. TML6, 58 (lines 552-592). Accordingly, the first intermediate value “*is*32*” is truncated by dividing it by first scale factor **1024**: ($\text{“is*32”}/1024$). Scale factor **1024** equals the sum of the weights ($32 \times 20 + 32 \times 20 - 32 \times 5 - 32 \times 5 + 32 \times 1 + 32 \times 1 = 1024$) used in the weighted sum. Claim 21, *supra*. Since *is*32* has a dynamic range of 0 to 261,120, dividing it by scale factor **1024** produces a dynamic range of 0 to 255, equal to the dynamic range of the unit pixels ($261,120/1024=255$). *Id.*; Ramchandran, ¶234.

Similarly, first intermediate value “*is*” for the (0/4,2/4) sub-pixel is truncated by dividing it by a first scale factor **1024**. Namely, it is divided by first scale factor **1024** when written to *mref*. TML6, 58 (lines 576-592). Scale factor **1024** equals the sum of the weights ($32*20+32*20-32*5-32*5+32*1+32*1=1024$) used in the weighted sum. Claim 21, *supra*. Since “*is*” has a dynamic range of 0 to 261,120, dividing it by scaling factor **1024** produces a dynamic range of 0 to 255, equal to the dynamic range of the unit pixels ($261,120/1024=255$). *Id.*; Ramchandran, ¶235.

The '273 patent confirms TML6 discloses the claimed truncation. EX-1001, 11:35-56; Ramchandran, ¶236.

u. Claim 24

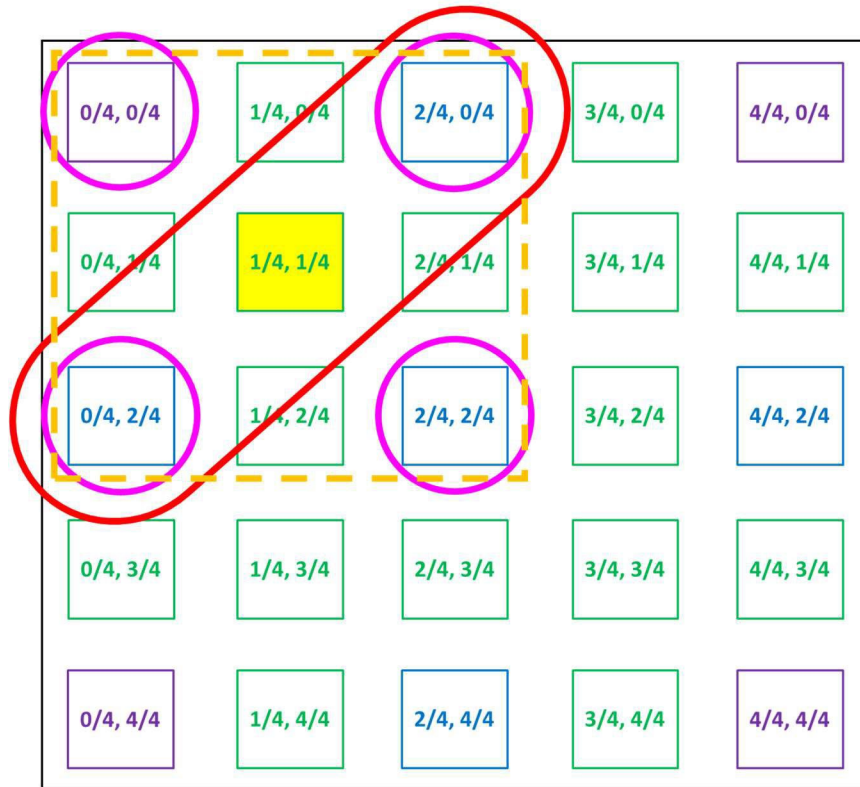
The TML6-Fandrianto combination teaches Claim 24's additional features. As discussed for 1[a], sub-pixels having co-ordinates with odd values of K and L include sub-pixels (1/4,1/4), (1/4,3/4), (3/4,1/4), and (3/4,3/4), highlighted below.



K=Odd, L=Odd

The TML6-Fandrianto combination interpolates highlighted (1/4,1/4) sub-pixels based on a predetermined choice of two weighted averages, one of which (below) includes values for sub-pixels (red; circled) at co-ordinates (2/4,0/4) and (0/4,2/4), stored in *mref*.

TML6 + Fandrianto

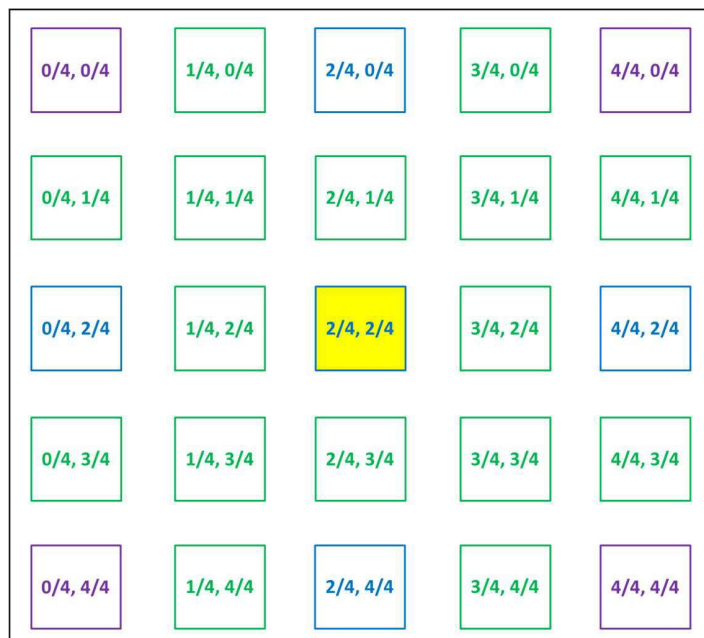


Id. These values are the truncated first intermediate values for the (2/4,0/4) and (0/4,2/4) sub-pixels. Claim 23, *supra*. Therefore, the TML6-Fandrianto combination uses a truncated first intermediate sub-pixel value when a sub-pixel having co-ordinates with odd values of K and L. 1[a], *supra*; Ramchandran, ¶¶237-238.

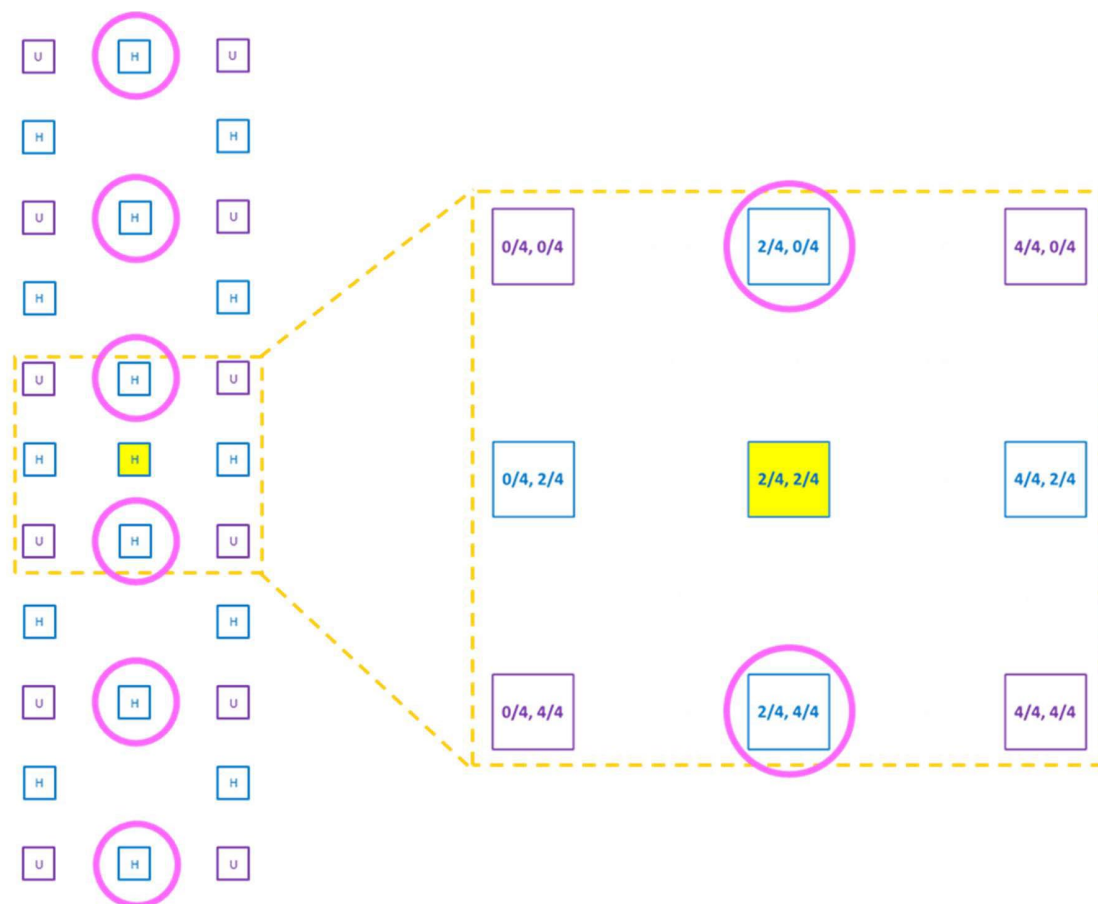
v. Claim 25

The TML6-Fandrianto combination teaches Claim 25's additional features. As discussed for 1[c], TML6 interpolates highlighted (2/4,2/4) sub-pixels (blue)

having co-ordinates $(K/2^N, L/2^N; N=2)$ with even values of K and L ($K=L=2$) using $(2/4, 0/4)$ sub-pixels from *img4Y_tmp* to produce second intermediate value “*is*”.



$(2/4, 2/4)$
K=2, L=2



First Weighted Sum

TML6, 57 (lines 563-592). This second intermediate value “*is*” has a dynamic range (0 to 261,120) of values equal to the dynamic range of the first intermediate values (0 to 261,120), which correspond to the (2/4,0/4) sub-pixels (blue) in *img4Y_tmp* from *imgY* (Claim 21, *supra*) used in the weighted sum multiplied by a value (1) equal to the sum of the respective weights used in the weighted sum. As highlighted below, this weighted sum (for the value “*is*”) has weights equal to the values from *ONE_FOURTH_TAP* divided by 32 ($20/32+20/32-5/32-5/32+1/32+1/32=1$). TML6, 3 (lines 28-34); Claim 21, *supra*; Ramchandran, ¶¶239-240.

Alternatively, the second intermediate value is the portion of “*is*” (within brown boundary below) before dividing by 32. This value has a dynamic range of “*is*” times 32 (0 to 8,355,840) of values equal to the dynamic range (0 to 261,120) of the first intermediate values—corresponding to the (2/4,0/4) sub-pixels (blue) in *img4Y_tmp* (Claim 23, *supra*)—used in the weighted sum multiplied by a value (32) equal to the sum of the respective weights used in the weighted sum. TML6, 3 (lines 563-592); Claim 21, *supra*. The weights for the second intermediate value are values from *ONE_FOURTH_TAP* (20+20-5-5+1+1=32). TML6, 3 (lines 28-34); Ramchandran, ¶241.

As shown below, the '273 patent confirms TML6 interpolates (2/4,2/4) sub-pixels (blue) in this manner.

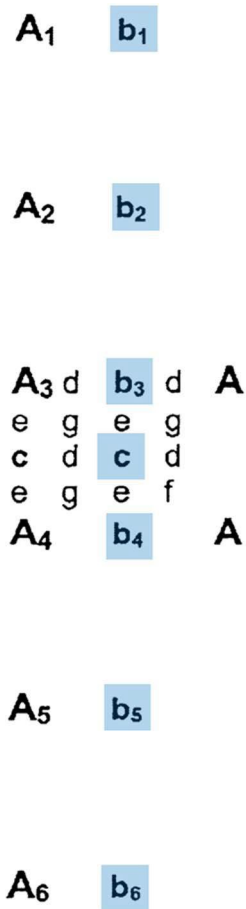


Figure 13(c)

EX-1001, Figure 13(c), 11:46-63; Ramchandran, ¶242.

w. Claim 26

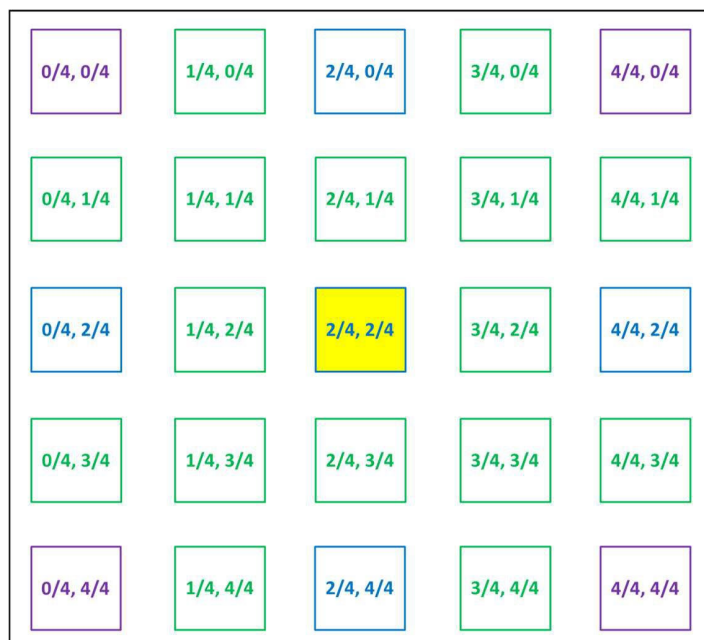
The TML6-Fandrianto combination teaches Claim 26's additional features. TML6 discloses the second intermediate value "*is*" for the (2/4,2/4) sub-pixels which is truncated by dividing it by a second scale factor 1024 and written to *mref*. Claim 25, *supra*; TML6, 58 (lines 552-592). The second scale factor 1024 is equal to the sum of the respective weights used in the weighted sum of Claim 21—1024 when

the value “***is**32**” constitutes the first intermediate values for the (2/4,0/2) sub-pixels—multiplied by the sum of the weights used in the weighted sum of Claim 25—“1” when the value “***is***” constitutes the second intermediate values for the (2/4,2/4) sub-pixels. *Id.*; 1[b]-1[c], *supra*. Since the second intermediate value “***is***” has a dynamic range of (0 to 261,120), dividing by the scale factor 1024 gives a dynamic range of 0 to 255 equal to the dynamic range of the unit pixels, 0 to 261,120 divided by 1024 (=255). *Id.*; Claim 20, *supra*; Ramchandran, ¶¶243-244.

The '273 patent confirms TML6 discloses the claimed truncation. EX-1001, 11:46-63; Ramchandran, ¶245.

x. Claim 28

The TML6-Fandrianto combination teaches Claim 28's additional features. As discussed for 1[c], in the combination the highlighted (2/4,2/4) sub-pixels (blue) (below) have co-ordinates ($K/2^N, L/2^N$; $N=2$) with even values of K and L (**$K=L=2$**). These highlighted sub-pixels are also interpolated as discussed above for Claim 25, except the (0/4,2/4) sub-pixels (blue) and values for sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions are used instead of (2/4,0/4) sub-pixels (blue) and corresponding values for sub-pixels having corresponding co-ordinates in immediately adjacent rectangular bounded regions.



(2/4,2/4)
K=2, L=2

Thus, as in Claim 25, the TML6-Fandrianto combination interpolates (2/4,2/4) sub-pixels using a weighted sum comprising first intermediate values for sub-pixels having co-ordinates $(K/2^N, L/2^N; N=2)$ with K equal to zero and L equal to an even value (**L=2**) and corresponding first intermediate values for sub-pixels having co-ordinates in immediately adjacent rectangular bounded regions to produce a second intermediate value having a dynamic range equal to the dynamic range of the first intermediate values used in said weighted sum, multiplied by a value equal to the sum of the respective weights used in said weighted sum, as claimed. Ramchandran, ¶¶246-247.

y. Claim 29

The TML6-Fandrianto combination teaches Claim 29's additional features. TML6 discloses "truncating the second intermediate value by mathematically dividing the second intermediate value by a second scale factor, said second scale factor having a value equal to the sum of the respective weights used in the weighted sum of Claim 21 multiplied by the sum of the respective weights used in the weighted sum [to interpolate the (2/4,2/4) sub-pixels], thereby forming a sub-pixel value with a dynamic range equal to the dynamic range of said pixels." Claim 26 *supra*. For similar reasons as discussed above, the TML6-Fandrianto combination also teaches using (0/4,2/4) sub-pixels to interpolate the (2/4,2/4) sub-pixel, as recited in Claim 28, and thus discloses this claim's additional features. Ramchandran, ¶¶248-249.

3. Independent Claim 31

Claim 31 is substantially the same as Claim 1 but is directed to $N=2$, which TML6 teaches. 1[pre][b], *supra*; Ramchandran, ¶250.

4. Claims 33, 36-49, 51, 53-59, and 61-62

a. Independent Claim 33

Claim 33 is substantively the same as Claim 1, except it is written in terms of "[a]n **interpolator**...configured to" perform the method of Claim 1. Ramchandran, ¶251.

If “interpolator” is limiting, TML6 and Fandrianto each disclose interpolating sub-pixels from other pixels and/or previously interpolated sub-pixels and are thus interpolators. Ramchandran, ¶¶252-254. TML6 discloses an interpolator for sub-pixel interpolation. TML6, 48 (line 3). TML6 includes a function that “[e]ncodes one frame” of video data. TML6, 48-50 (lines 33-162). This function calls the functions *oneforthpix()* and *oneforthpix_2()* to interpolate sub-pixels for the frame of video data. TML6, 49 (lines 102-109), 57-60 (lines 538-719); Ramchandran, ¶¶134, 47-108.

Fandrianto describes ALU 154 within “programmable architecture 100” that receives pixel data and performs pixel averaging arithmetic to generate intermediate pixel values during half-pixel and quarter-pixel interpolation processes. Fandrianto, 3:55-4:6, 4:38-42, 4:56-64, 6:26-32, 6:61-65, 9:4-32, Figure 2. These operations involve averaging pixel values retrieved from memory and storing the resulting interpolated values, demonstrating the functionality of an interpolator. Ramchandran, ¶253.

TML6 and Fandrianto thus disclose an “interpolator.” Ramchandran, ¶¶251-254.

b. Claims 36-49, 53-59, and 61-62

These claims are substantively the same as Claims 4-17, 20-26 and 28-29, except they are written in terms of “[a]n **interpolator**...configured to” perform the

respective method step(s) of those claims. As discussed for Claim 33, TML6 and Fandrianto disclose an interpolator. Ramchandran, ¶¶255-258.

c. Claim 51

As discussed for Claim 1, TML6 discloses the case where $N=2$. Ramchandran, ¶259. As discussed for Claim 33, TML6 and Fandrianto both disclose an interpolator. Ramchandran, ¶259.

5. Claims 64-67

a. Independent Claim 64

Claim 64 is substantively the same as Claim 1, except it is written in terms of “[a] **video encoder comprising an interpolator**...configured to” perform the method of Claim 1. Ramchandran, ¶260.

If “video encoder” is limiting, TML6 and Fandrianto disclose it. TML6 encodes/decodes video data. *See* TML6, 155 (line 5), 69-70 (lines 3, 86), 1 (lines 3, 18). VCEG developed video coding standards for implementation in hardware or software coders for, e.g., videoconferencing and video broadcast. VCEG-L45, §8; Ramchandran, ¶261; Wenger, ¶¶15, 19-21. Fandrianto also discloses a video encoder/decoder. Fandrianto, 1:16-67, Figure 2. A POSITA would have thus understood TML6 and Fandrianto teach a “video encoder.” Ramchandran, ¶¶260-263, 105-110.

If “interpolator” is limiting, a POSITA would have understood that TML6 and Fandrianto both disclose “[a] video coder...comprising an interpolator” as discussed for Claim 33. Ramchandran, ¶252.

b. Claim 65

Claim 65 is directed to “[a] **codec comprising a video encoder** according to claim 64 **and a video decoder** comprising an interpolator...configured to” perform the method of Claim 1. TML6 and Fandrianto teach a “codec” comprising a video encoder and decoder. Ramchandran, ¶264. A codec is a “hardware- or software-based video compression/decompression adapter[.]” Riley, 9, 12; CRC-Dictionary, 28. As discussed for Claim 64, TML6 and Fandrianto both encode/decode (compress/decompress) video data, and thus disclose a video codec. Ramchandran, ¶¶264-268.

c. Claim 66

The ’273 patent describes a “communications terminal” as “comprising a user interface,...processor[, and]...transmitting[/receiving] block[s]” in addition to “a video coder.” EX-1001, 18:11-14. If “[a] communications terminal comprising a video encoder” is limiting, TML6 and Fandrianto disclose it. TML6 is designed for a video encoder/decoder in a communications terminal because VCEG developed TML6 for use in applications that receive packetized data over IP networks, which require communications terminals. VCEG-L45, 34-36. Furthermore, Fandrianto’s

technique is designed for implementation by a processor in digital television broadcast and video telephony applications, which require communications terminals. Fandrianto, 1:12-41, 3:55-4:17, 11:32-14:40, 20:38-51, Figure 2; Ramchandran, ¶269. A POSITA would have thus implemented the TML6-Fandrianto combination in a “communications terminal,” as recited. Ramchandran, ¶269.

d. Claim 67

The TML6-Fandrianto combination teaches Claim 67’s additional features. TML6 was developed for applications that receive data over IP networks. Claim 64, *supra*. Fandrianto’s interpolation technique is designed for digital television broadcast—over wireless radio waves received by wireless terminals—and video telephony systems, which likewise require communications terminals. Fandrianto, 1:12-41; Ramchandran, ¶270. Wireless communication terminals that sent/received coded video were well-known by the earliest priority date. Fandrianto, 18:30-34; VCEG-L45, 34-36; Nakaya, 11:64-12:47, 13:47-50, Figure 14; Riley, 7-14.

6. Claims 68-70

a. Independent Claim 68

Claim 68 is substantively the same as Claim 64 except it is directed to a “**still image** encoder” rather than a “**video** encoder.” Video comprises a series of still images played in rapid succession to create the illusion of motion. Ramchandran,

¶271. Video encoders compress/encode this series of still images for efficient transmission and storage. Ramchandran, ¶¶271-272.

TML6 expressly teaches video formats, e.g., QCIF and CIF. TML6, 1. A POSITA familiar with such formats would have understood the underlying principles of image resolution and frame composition such that they could easily apply these principles to extract and save individual frames as still images in formats like JPEG or PNG, since video frames are essentially a sequence of still images. The conversion between video frames and still images was a common practice, facilitated by available software tools and hardware capabilities at that time. Therefore, a still image encoder would have been within the expertise of a POSITA. Ramchandran, ¶272.

b. Claim 69

Claim 69 is substantively the same as Claim 65, except it is directed to a “**still image** encoder” rather than a “**video** encoder.” The TML6-Fandrianto combination teaches Claim 69 for the same reasons as Claim 68. Ramchandran, ¶273.

c. Claim 70

Claim 70 is substantively the same as Claim 66, except it is directed to a “**still image** encoder” rather than a “**video** encoder.” The TML6-Fandrianto combination teaches Claim 70 for the same reasons as Claim 69. Ramchandran, ¶274.

7. Claims 71-72

a. Independent Claim 71

Claim 71 is substantively the same as Claim 1, except it is directed to “[a] **video decoder comprising an interpolator...configured to**” perform the method of Claim 1. If limiting, TML6 and Fandrianto disclose these limitations as discussed for Claim 64. Ramchandran, ¶¶275-276.

b. Claim 72

The TML6-Fandrianto combination teaches Claim 72 for the same reasons as Claim 66. Ramchandran, ¶¶277-278.

8. Claims 73-74

a. Independent Claim 73

Claim 73 is substantively the same as Claim 68, except it is written in terms of “[a] still image **decoder**” rather than “[a] still image **encoder**.” Ramchandran, ¶279. The TML6-Fandrianto combination teaches Claim 73 for the same reasons as Claims 68-70. Ramchandran, ¶279.

b. Claim 74

The TML6-Fandrianto combination teaches Claim 74 for the same reasons as Claims 68-72. Ramchandran, ¶280.

9. Independent Claim 75

Claim 75 is substantively the same as Claim 1, except it is written in terms of “[a]n **interpolator...comprising[] circuitry** configured to” perform the steps of

Claim 1. Ramchandran, ¶281. TML6 and Fandrianto teach an “interpolator.” Claim 33, *supra*. The ’273 patent confirms that TML6 is source code created to execute on a processor requiring circuitry. EX-1001, 12:65-13:17 (“TML6... requires more silicon area in ASICs and requires more computations in some CPUs”). TML6 thus discloses “circuitry configured to interpolate.” Ramchandran, ¶309.

Fandrianto discloses an apparatus including circuitry operable to interpolate sub-pixels. Fandrianto 1:61-67 (“effective video coding requires integration of...integrated circuit technology...”); *see id.*, Figures 2-5, 2:48-58, 3:55-4:53, 5:26-7:2, 7:27-10:58; Ramchandran, ¶310.

The TML6-Fandrianto combination thus discloses “circuitry configured to” interpolate the claimed sub-pixels. Ramchandran, ¶¶281-324.

10. Claims 76-78

a. Independent Claim 76

Claim 76 is substantially the same as Claim 75 but is directed to the case where $N=2$, which TML6 teaches. Ramchandran, ¶325.

b. Claims 77-78

As discussed for Claims 64-65, if limiting, TML6 and Fandrianto teach a “video encoder” and “communications terminal.” Ramchandran, ¶326.

11. Independent Claim 82

Claim 82 is substantively the same as Claim 1, except it is written in terms of “[a] **computer program**...embodied on a **computer readable storage medium**, the computer program...comprising **computer program code**” for performing the method of Claim 1. Ramchandran, ¶327.

If “[a] computer program [comprising computer program code]...embodied on a computer readable storage medium” is limiting, TML6 and Fandrianto disclose it. TML6 is source code and thus satisfies this limitation. *See* TML6.; Ramchandran, ¶327. Furthermore, Fandrianto discloses that its architecture’s controller “is operated by microcode, but may be software controlled if desired.” Fandrianto, 11:38-45, Figure 2.

12. Independent Claim 83

Claim 83 is substantively the same as Claim 33, except it is written in terms of “[a] **network apparatus** comprising an interpolator...configured to” perform the method of Claim 1. Ramchandran, ¶328.

TML6 and Fandrianto teach an “interpolator.” Claim 33, *supra*. If “network apparatus” is limiting, it was well-known to implement a sub-pixel interpolator in hardware or software within a network of terminals that send/receive coded video. Fandrianto, 18:30-34; VCEG-L45, 34-36; Nakaya, 11:64-12:47, 13:47-50, Figure

14; Riley, 7-14. The TML6-Fandrianto combination discloses these limitations as discussed for Claims 33, 66-67, and 75. Ramchandran, ¶¶328-329.

B. Ground 2: Claims 32, 52, and 79-81 are Obvious over TML6 in View of Fandrianto and the Knowledge of a POSITA

1. Independent Claim 32

Claim 32 is substantially the same as Claim 31 but is directed to “**eighth** resolution” rather than “quarter resolution...interpolation.” The difference in resolution is reflected in the coordinate notations and sub-pixels recited in Claim 32. Ramchandran, ¶330.

A POSITA would have been motivated to modify the TML6-Fandrianto combination for eighth pixel resolution because the prior art expressly motivates the modification. *See KSR*, 550 U.S. at 418-419. By 2001, sub-pixel interpolation was a well-known technique for enhancing the resolution of digital images. Ramchandran, ¶¶330-331. It was specifically known that eighth resolution sub-pixel accuracy offered a desirable balance between improved image quality and manageable computational complexity. For instance, the draft successor to TML6—TML8, published in July 2001—implemented 1/8 pixel accuracy “[f]or a higher complexity or higher coding efficiency profile.” VCEG-N10, 1, 14.

A POSITA would have recognized that setting N equal to 3—instead of 2, as disclosed in TML6, *see* 1[pre][b]—would achieve eighth pixel resolution in the

TML6-Fandrianto system. Ramchandran, ¶¶330, 332. As Dr. Ramchandran explains, $N=3$ was specifically known to be a desirable value to enhance the resolution and smoothness of an approximated function. Ramchandran, ¶334 (citing Press, 106-107).

A POSITA would have recognized setting $N=3$ as the use of a known technique (higher-order interpolation) to improve similar devices (digital video systems) in the same way (increasing resolution and smoothness using sub-pixel interpolation). Ramchandran, ¶¶333, 337. Modifying the TML6-Fandrianto combination to achieve eighth pixel resolution would have been the application of a known technique (1/8 interpolation) to a known device (video encoder/decoder) ready for improvement to yield predictable results (video coding system with improved video fidelity). *KSR*, 550 U.S. at 417; Ramchandran, ¶336.

A POSITA would have had a reasonable expectation of success with predictable results because setting $N=3$ in the TML6-Fandrianto combination would have been a trivial modification within the skill of a POSITA, and the resulting system would have worked in the same way. Ramchandran, ¶336. Furthermore, eighth resolution interpolation was achievable on hardware available in 2001, making it a practical choice for high-resolution displays and imaging applications. Ramchandran, ¶¶335-336.

2. Claim 52

The TML6-Fandrianto combination in view of the knowledge of a POSITA teaches the additional limitations of this claim as discussed for Claims 32-33. Ramchandran, ¶338.

3. Claims 79-81

a. Independent Claim 79

Claim 79 is substantially the same as Claim 33 but specifies “[a]n interpolator for **eighth resolution...interpolation**” rather than Claim 33’s “interpolator for sub-pixel...interpolation.” The specified resolution is reflected in both the coordinate notations and the specific sub-pixels recited in Claim 79. Ramchandran, ¶339. As discussed for Claims 32-33, the TML6-Fandrianto combination in view of the knowledge of a POSITA teaches “[a]n interpolator for eighth resolution sub-pixel...interpolation.” Ramchandran, ¶339.

b. Claim 80

The TML6-Fandrianto combination in view of the knowledge of a POSITA teaches the additional limitations of this claim as discussed for Claims 32-33 and 68-70. Ramchandran, ¶340.

c. Claim 81

The TML6-Fandrianto combination in view of the knowledge of a POSITA teaches the additional limitations of this claim as discussed for Claims 32-33 and 68-70. Ramchandran, ¶341.

IX. NO BASIS FOR DISCRETIONARY INSTITUTION DENIAL

A. General Plastic

This is Amazon’s first petition for IPR of the ’273 patent. The ’273 patent was challenged in four prior IPRs filed in 2020 by Lenovo (United States) Inc. (“Lenovo”): IPR2021-00040, IPR2021-00041, IPR2021-00042, and IPR2021-00043. Although *General Plastic* addressed IPR petitions challenging the same patent filed by “the same petitioner,” the Board in *Valve Corp. v. Elec. Scripting Prods., Inc.*, IPR2019-00062, Paper 11 (Apr. 2, 2019) (precedential) extended *General Plastic* factor 1 to include IPR petitions challenging the same patent filed by petitioners sharing a “significant relationship...with respect to [the challenged patent].” *Valve* at 9-10.

Amazon has never had a “significant relationship” with Lenovo regarding the ’273 patent. See *Ford Motor Co. v. Neo Wireless LLC*, IPR2023-00763, Paper 28, 7 (Mar. 22, 2024). In *Ford*, the Director vacated and remanded the Board’s discretionary denial because the first and second petitioners’ “relationship...[was] premised on the allegation that they each infringe the same patent, but with different

allegedly infringing products and in different district court proceedings.” *Ford* at 10. The same is true of Amazon and Lenovo. Patent Owner filed suit against Lenovo in the Eastern District of North Carolina on September 25, 2019. EX-1019. Patent Owner and Lenovo stipulated to dismissal of that case on April 12, 2021. EX-1020. Patent Owner filed suit against Amazon in the District of Delaware on October 31, 2023—over two-and-a-half years after dismissal of the Lenovo case—alleging infringement of different products. EX-1010. Amazon and Lenovo thus lack the “closely aligned interests with respect to the [’273] patent” necessary for a “significant relationship” to exist. *Ford* at 9-10.

“[W]here, as here, the first and second petitioners are neither the same party, nor possess a significant relationship under *Valve, General Plastic* factor one necessarily outweighs the other *General Plastic* factors.” *Videndum Production Solutions, Inc. v. Rotolight Ltd.*, IPR2023-01218, Paper 12, 6 (Apr. 19, 2024). Moreover, there have been no communications, written or oral, between Amazon and Lenovo relating to the ’273 patent or the filing of this Petition, including its financing, preparation, editing, review, approval, or filing. No individuals acting for or on behalf of Lenovo participated or assisted in any way with any of these activities. There have been no payments or agreements between Lenovo and Amazon, directly or indirectly, in connection with this Petition or the ’273 patent.

The Board “frequently decline[s] to exercise discretionary denial under *General Plastic* where there is no ‘significant relationship’ between parties challenging the same patent.” *Samsung Bioepis Co. Ltd. v. Regeneron Pharms., Inc.*, IPR2023-00442, Paper 10 at 46 (July 19, 2023). The Board should do so here.

B. *Fintiv*

There are two co-pending district court proceedings involving the ’273 patent. Both were initiated in late 2023, and no schedules have been entered. Any trial would therefore be expected well after the FWD deadline, likely December 2025. Further, it is unclear whether any overlap exists between this Petition and issues that might be raised in the district court, as the parties have not yet exchanged preliminary infringement or invalidity contentions. Discretionary denial under *Fintiv* is thus unjustified.

C. §325(d)

The Board should likewise not exercise its discretion under §325(d) to deny institution. *See Becton, Dickinson & Co. v. B. Braun Melsungen AG*, IPR2017-01586, Paper 8, 17-18 (Dec. 15, 2017) (precedential); *Advanced Bionics, LLC v. Med-El Elektromedizinische Geräte GmbH*, IPR2019-01469, Paper 6, 9-11 (Feb. 13, 2020) (precedential).

The Examiner did not substantively consider either TML6 or Fandrianto during prosecution. Although the ’273 patent accurately summarizes certain

portions of the TML6 sub-pixel interpolation scheme (EX-1001, 11:25-13:27), the TML6 source code relied on herein (EX-1005) includes far greater coverage of the components and functionalities of the TML6 codec and was thus not considered by the Examiner. *See Westinghouse Air Brake Techs. v. Siemens Indus., Inc.*, IPR2017-02044, Paper 12, 34 (Feb. 7, 2018) (declining under §325(d) to deny petition that relied on a prior art reference that was merely summarized in the Background section of the challenged patent). The '273 patent itself indicates TML6 was not considered substantively because TML6 is not listed among the numerous references cited. EX-1001, (56). There is also no evidence that TML6 was provided to the Office by the applicant, such as on an IDS. *See* EX-1002.

TML6 is also not “substantially the same [as] prior art or arguments previously...presented to the Office” under §325(d). The '273 patent summarizes certain aspects of TML6 with reference to Figures 13(a)-(c). EX-1001, 11:25-13:27. But the '273 patent's two-column description lacks the detail found in TML6's nearly 10,000 lines of code, e.g., how the *oneforthpix()* function interacts with the *imgY*, *img4Y_tmp*, and *mref* data structures. Claim 1, *supra*. Accordingly, TML6 was not considered by the Examiner.

In any event, denial under §325(d) is unwarranted because the Examiner did not consider the specific combination of TML6 and Fandrianto. *Volkswagen Grp. of Am., Inc. v. Mich. Motor Techs. LLC*, IPR2020-00169, Paper 8, 26 (June 2, 2020).

Fandrianto was cited in an IDS but never applied against the claims nor discussed by the Examiner. EX-1002, 206. Such a reference does not weigh in favor of exercising the Board's discretion under §325(d) to deny a petition. *See Zip-Top LLC v. Stasher, Inc.*, IPR2018-01216, Paper 14, 35-36 (Jan. 17, 2019) (“[M]ere citation to a reference by the Examiner does not establish that the Examiner substantively considered the merits of [the reference].”); *Shenzhen Zhiyi Tech. Co. v. iRobot Corp.*, IPR2017-02137, Paper 9, 9-10 (Apr. 2, 2018).

X. CONCLUSION

Inter partes review of the Challenged Claims is respectfully requested.

XI. MANDATORY NOTICES

A. Real Party in Interest

Petitioner identifies themselves as real parties in interest.

B. Related Matters

To the best of Petitioner's knowledge, the '273 patent has been involved in the following related matters:

- *Nokia Techs. Oy v. Lenovo (Shanghai) Elecs. Tech. Co.*, No. 5:19-cv-00427 (E.D.N.C.)
- *Nokia Techs. Oy v. Amazon.com, Inc.*, No. 1:23-cv-01236 (D. Del.)
- *Nokia Techs. Oy v. HP Inc.*, No. 1:23-cv-01237 (D. Del.)
- *Nokia Techs. Oy v. Apple Inc.*, No. 2-16-cv-01440 (E.D. Tex.)

- IPR2021-00040
- IPR2021-00041
- IPR2021-00042
- IPR2021-00043

C. Notice of Counsel and Service Information

Lead Counsel	Backup Counsel
Jessica Kaiser, Reg. No. 58,937 PERKINS COIE LLP 1900 Sixteenth Street, Suite 1400 Denver, CO 80202 Tel: (303) 291-2300 Kaiser-ptab@perkinscoie.com	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 Jonathan R. Carter, Reg. No. 75,145 PERKINS COIE LLP 1155 Avenue of the Americas 22nd Floor New York, NY 10036 Tel: (212) 261-6826 Fax: (212) 977-1649 Carter-ptab@perkinscoie.com

Petitioner consents to electronic service at the following email address:
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: July 5, 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. 8,036,273, 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 13,945. 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: July 5, 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. 8,036,273 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:

CPA GLOBAL
Clarivate
3133 W. Frye Rd.
STE 400
Chandler, AZ 85226

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: July 5, 2024