

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMAZON.COM, INC. AND AMAZON.COM SERVICES LLC,
Petitioners

v.

INTERDIGITAL, INC.,
Patent Owner.

IPR2026-00195
U.S. Patent No. 10,250,877

DECLARATION OF JOSEPH P. HAVLICEK, PH.D.

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

Date: 8 January 2026

By: 

Joseph P. Havlicek, Ph.D.

TABLE OF CONTENTS

I.	BACKGROUND AND QUALIFICATIONS.....	1
A.	Educational Background	1
B.	Professional Experience	2
C.	Publications	6
II.	MATERIALS CONSIDERED	7
III.	LEGAL STANDARDS.....	7
IV.	TECHNOLOGY BACKGROUND.....	12
A.	Video Compression.....	12
1.	Resampling/Resizing	15
2.	Block Coding	19
3.	Predictive Coding.....	20
4.	Combining Compression Methods	29
B.	Standards Development	30
V.	'877 PATENT.....	32
A.	Specification.....	32
B.	Prosecution History.....	36
VI.	PRIOR ART	38
A.	Nakagawa (Ex-1007)	38
B.	Sita (Ex-1008)	40
C.	Wiegand (Ex-1009).....	42
D.	Davies (Ex-1010)	43
E.	Fimoff (Ex-1011)	45
VII.	LEVEL OF ORDINARY SKILL.....	50
VIII.	CLAIM CONSTRUCTION.....	51
IX.	GROUND	52
A.	Ground 1: Obviousness over Nakagawa in view of Sita	52
1.	Claim 1	52

TABLE OF CONTENTS
(continued)

	Page
a. 1[pre].....	52
b. 1[a]	61
c. 1[b].....	71
d. 1[c]	73
e. 1[d].....	75
f. 1[e]	91
g. 1[f].....	92
2. Claim 4.....	96
a. 4[pre].....	96
b. 4[a]	97
c. 4[b]	97
d. 4[c]	97
e. 4[d].....	98
f. 4[e]	99
g. 4[f]	99
3. Claim 7.....	99
a. 7[pre].....	99
b. 7[a]	100
c. 7[b].....	101
d. 7[c]	101
e. 7[d].....	101
f. 7[e]	101

TABLE OF CONTENTS
(continued)

	Page
g. 7[f]	102
h. 7[g]	102
4. Claim 8	102
a. 8[pre]	102
b. 8[a]	102
c. 8[b]	103
d. 8[c]	103
e. 8[d]	103
f. 8[e]	103
g. 8[f]	104
h. 8[g]	104
B. Ground 2: Obviousness over Wiegand in view of Davies, and Fimoff	104
1. Claim 1	104
a. 1[pre]	104
b. 1[a]	114
c. 1[b]	123
d. 1[c]	126
e. 1[d]	127
f. 1[e]	130
g. 1[f]	136
2. Claim 4	139

TABLE OF CONTENTS
(continued)

	Page
a. 4[pre].....	145
b. 4[a]	146
c. 4[b].....	147
d. 4[c]	148
e. 4[d].....	150
f. 4[e]	150
g. 4[f].....	150
3. Claim 7.....	151
a. 7[pre].....	151
b. 7[a]	152
c. 7[b].....	152
d. 7[c]	152
e. 7[d].....	153
f. 7[e]	153
g. 7[f].....	153
h. 7[g].....	154
4. Claim 8.....	154
a. 8[pre].....	154
b. 8[a]	154
c. 8[b].....	155
d. 8[c]	155
e. 8[d].....	155

TABLE OF CONTENTS
(continued)

	Page
f. 8[e]	155
g. 8[f]	156
h. 8[g]	156
C. Ground 3: Obviousness over Wiegand in view of Davies and Sita.....	156
1. 1[a], 4[a], 7[b], and 8[b].....	157
2. 1[d]-[e], 4[d]-[e], 7[e]-[f], 8[e]-[f]	160
3. 1[f], 4[f], 7[g], and 8[g]	161
X. SUMMARY AND ADDITIONAL REMARKS.....	161

I, Dr. Joseph P. Havlicek, do hereby declare:

I. BACKGROUND AND QUALIFICATIONS

1. I have been retained by Petitioners Amazon.com, Inc. and Amazon.com Services LLC (“Amazon” or “Petitioner”) to provide my opinions regarding claims 1, 4, 7, and 8 of U.S. Patent No. 10,250,877 (“the ’877 Patent,” Ex-1001).

2. I am currently Gerald Tuma & Williams Companies Foundation Presidential Professor of Electrical and Computer Engineering at the University of Oklahoma.

3. Available at Ex-1004 is a copy of my current curriculum vitae, which provides further details about my academic and professional background and qualifications.

4. I am being compensated for my services and reimbursed for reasonable expenses that I may incur while working on this matter. I have no financial interest in the outcome of this matter or any other matter that may exist between Petitioners and Interdigital, Inc. (“Interdigital” or “Patent Owner”). This compensation does not depend in any way on the conclusions I reach.

A. Educational Background

5. I received a Bachelor of Science Degree in electrical engineering with minors in mathematics and computer science from Virginia Tech in 1986. I also received a Master of Science degree in electrical engineering from Virginia Tech in

1988. I received a Ph.D. degree in Electrical and Computer Engineering from the University of Texas at Austin in 1996. My Ph.D. research was in the field of image processing.

B. Professional Experience

6. From December 1984 to May 1987, I was a software engineer at Management Systems Laboratories, Blacksburg, VA. My main job responsibilities included developing software for nuclear materials management under contract with the United States Department of Energy.

7. From June 1987 to January 1997, I was an electrical engineer at the United States Naval Research Laboratory. For the period of June 1987 through August 1989, I was an on-site contractor affiliated with SFA, Inc., Landover, Maryland. From August 1989 through January 1997, I was a regular government employee. I was on leave without pay from August 1987 through July 1988 while completing my Master of Science Degree. I was also on leave without pay for much of the period from August 1990 through January 1997 while I completed my Ph.D. degree. My main job responsibilities at the United States Naval Research Laboratory included designing digital and analog circuits to process real-time video signals and designing and implementing target detection, tracking, and identification algorithms. I developed missile warning receivers for Navy aircraft including the Navy's first two-color infrared system. I was a recipient of the 1990 Department of the Navy

Award of Merit for Group Achievement for this work. The production version of this system was deployed on Navy helicopters and saw combat in Afghanistan and Iraq.

8. From January 1993 through December 1993, I was an on-site contractor at International Business Machines (IBM) Corporation, Austin, TX. My main job responsibilities included designing and implementing image compression and decompression algorithms (CODECs) for IBM products.

9. Since January 1997, I have been a regular faculty member in the School of Electrical and Computer Engineering at the University of Oklahoma, Norman, OK. I was an Assistant Professor from January 1997 through June 2002. I was promoted to the rank of Associate Professor and granted tenure in July 2002. I was promoted to the rank of Professor in July 2007. I was appointed to the Williams Companies Foundation Presidential Professorship in April 2009. In April 2017, I was appointed to the Gerald Tuma Presidential Professorship. My main job responsibilities at the University of Oklahoma include conducting academic research in electrical and computer engineering, teaching graduate and undergraduate courses in electrical and computer engineering, and performing professional and institutional service.

10. I am a member of several professional societies and organizations, including the Institute of Electrical and Electronics Engineers (IEEE), the IEEE

Signal Processing Society, the IEEE Computer Society, and the IEEE Intelligent Transportation Society. I am a Senior Member of the IEEE. From November 2015 through February 2018, I served as a Senior Area Editor for the IEEE Transactions on Image Processing. I was formerly an Associate Editor for the IEEE Transactions on Image Processing from December 2010 through October 2015. I served as a Technical Area Chair for the IEEE International Conference on Image Processing in the area of Image & Video Analysis, Synthesis, and Retrieval (2012, 2013) and have served on the organizing committee of that conference (2007). I have also served as a Technical Area Chair for the IEEE International Conference on Acoustics, Speech, and Signal Processing in the area of Image, Video, and Multidimensional Signal Processing (2012-2014).

11. For over 35 years, I have conducted research and taught classes in the field of image processing and image analysis. My main scholarly contributions have been in the areas of modulation domain image models and image processing (AM-FM image models), video target tracking, and distributed control of video networks for intelligent transportation systems. I have supervised or co-supervised 12 Ph.D. students to completion and I am currently supervising three Ph.D. students. I have been a member of 68 additional doctoral-dissertation committees. I have supervised 28 M.S. students to completion and I am currently supervising one additional M.S. students. I have served as a committee member on 73 additional M.S. committees.

12. I teach a variety of courses at the University of Oklahoma, including the required junior-level Signals and Systems course ECE 3793 (taught 21 times), the graduate level Digital Image Processing course ECE 5273 (taught 26 times), and the graduate level Digital Signal Processing course ECE 5213 (taught 19 times). I have received a number of teaching awards, including the University of Oklahoma College of Engineering Outstanding Faculty Advisor Award (2005-2006) and the University of Texas Engineering Foundation Award for Exemplary Engineering Teaching while Pursuing a Graduate Degree (1992).

13. Since joining the University of Oklahoma in January 1997, I have been Principal Investigator or Co-Principal Investigator on over 115 externally funded grants and contracts with a total value of \$27,801,061. My main research contributions have been in the areas of signal, image, and video processing, video target tracking, and intelligent transportation systems. I have been author or coauthor on over 130 scholarly publications in these areas. I was a recipient of the 1990 Department of the Navy Award of Merit for Group Achievement for my work in video target tracking. My research group at the University of Oklahoma originated the Virtual Traffic Management Center concept featured in a December 2014 FHWA technical report (Guidelines for Virtual Transportation Management Center Development) and a November 2014 FHWA national webinar with the same title. I am co-founder and director of the University of Oklahoma Center for Intelligent

Transportation Systems. Under my supervision, the Center has collaborated with the Oklahoma Department of Transportation since 1998 to design and implement the Oklahoma Statewide Intelligent Transportation System, including a geographically distributed video network that is currently deployed on major highways and interstates across the entire State of Oklahoma.

C. Publications

14. A complete listing of my publications is found in my curriculum vitae (Ex-1004). I highlight some of the publications relevant to the subject matter of the '877 Patent below.

15. I have published numerous peer reviewed scholarly journal articles, book chapters, and conference papers, including 24 journal articles, 9 book chapters, and 100 conference papers; the following are representative:

- a. J.P. Havlicek, T.N. Arian, H. Soltani, T. Przebinda, and M. Özeydin, "A preliminary case for Hirschman transform video coding," in Proc. IEEE Southwest Symp. Image Anal. & Interp., Santa Fe, NM, Mar. 29-31, 2020, pp. 104-107.
- b. E. Vorakitolan, J.P. Havlicek, R.D. Barnes, and A.R. Stevenson, "Simple, Effective Rate Control for Video Distribution in Heterogeneous Intelligent Transportation System Networks," in Proc. IEEE Southwest Symp. Image Anal. & Interp., San Diego, CA, Apr. 6-8, 2014, pp. 37-40.
- c. V. DeBrunner, J.P. Havlicek, T. Przebinda, and M. Özeydin, "Entropy- Based Uncertainty Measures for $L_2(R^n)$, $l_2(Z)$, and $l_2(Z/NZ)$ with a Hirschman Optimal Transform for $l_2(Z/NZ)$," IEEE Trans. Signal Process., vol. 53, no. 8, pp. 2690-2699, Aug. 2005.

- d. A.C. Bovik, J.P. Havlicek, M.D. Desai, and D.S. Harding, "Limits on Discrete Modulated Signals," *IEEE Trans. Signal Process.*, vol. 45, no. 4, pp. 867- 879, Apr. 1997.
- e. J.P. Havlicek and A.C. Bovik, "Image Modulation Models," in *Handbook of Image and Video Processing*, A.C. Bovik, ed., Communications, Networking, and Multimedia Series by Academic Press, San Diego, 2000, pp. 305- 316.
- f. P.C. Tay, J.P. Havlicek, S.T. Acton, and J.A. Hossack, "Properties of the magnitude terms of orthogonal scaling functions," *Digital Signal Process.*, vol. 20, no. 5, pp. 1330-1340, Sep. 2010.
- g. O. Alkhoul, V. DeBrunner, and J. Havlicek, "Hirschman Optimal Transform (HOT) DFT Block LMS Algorithm," in *Adaptive Filtering*, L. Garcia, ed., ISBN: 978-953-307-158-9, InTech, Sep. 2011, pp. 135-152.
- h. V. DeBrunner, M. Özaydın, T. Przebinda, and J. Havlicek, "The optimal solutions to the continuous- and discrete-time versions of the Hirschman uncertainty principle," in *Proc. IEEE Int'l. Conf. Acoust., Speech, Signal Process.*, Istanbul, Turkey, Jun. 5-9, 2000, vol. 1, pp. 81-84.

II. MATERIALS CONSIDERED

16. In developing my opinions below relating to the '877 Patent, I have considered all exhibits cited herein, including the '877 Patent, the '877 file history, and all of the relevant prior art.

III. LEGAL STANDARDS

17. Counsel has informed me of the legal principles that apply in determining whether patent claims are unpatentable for obviousness. Paragraphs below beginning with "I understand" or "I have been informed" capture my

understanding of the law as provided to me by counsel. The other paragraphs are my opinion.

18. I understand that a claim may be unpatentable as obvious under § 103(a) if the subject matter described by the claim as a whole would have been obvious to a hypothetical “person of ordinary skill in the art” (“POSITA”) in view of a prior-art reference, or combination of prior-art references at the time the claimed invention was made.

19. I understand that “the time the claimed invention was made” is often referred to as the priority or critical date.

20. I understand that, before considering the prior art, the claims are first construed to determine their scope. I understand that the claims are read from the perspective of POSITA at the time of the alleged invention. I understand that the words of a claim are generally given their ordinary and customary meaning as understood by a POSITA in light of the entire patent specification in which they appear, as well as the prosecution history of the patent.

21. I understand, however, that the words of a claim may be given a meaning other than their plain and ordinary meaning if: (1) a patentee expressly and unambiguously defined a term in a way that is different from its plain meaning; or (2) a patentee “disclaims” the full scope of a claim term’s plain meaning, for

instance, by making clear and unambiguous statements during prosecution that a claim term does or does not encompass certain features.

22. I understand that, after construing the claims, the teachings of the prior art are considered. I understand that the prior art includes references that were known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the date the invention was allegedly made by the patent holder. For a printed publication to qualify as prior art, I understand that the Petitioner must demonstrate that the publication was disseminated or otherwise sufficiently accessible to the public. However, I also understand that a POSITA is assumed to be aware of all relevant prior art in the field of endeavor covered by the patent-at-issue and all analogous prior art.

23. I understand that although the ultimate question of the obviousness of a claimed invention is a legal determination, obviousness is based on several factual inquiries, including: the scope and content of the prior art, the differences between the claimed subject matter and the prior art, the level of ordinary skill in the art at the time of the invention, and any “objective indicia” or “secondary considerations” of non-obviousness, which must all be considered.

24. I understand that, in determining the scope and content of the prior art, a reference is considered relevant prior art if it falls within the field of the inventor’s endeavor, or if it is reasonably pertinent to the particular problem that the inventor

was trying to solve. A reference is reasonably pertinent if it logically would have commended itself to an inventor's attention in considering his problem.

25. I understand that, to assess the differences between prior art and the claimed subject matter, the claimed invention must be considered as a whole. I understand that this involves showing that a POSITA, confronted by the same problems as the inventor and with no knowledge of the claimed invention, would have identified the elements in the prior art and combined them in the claimed manner.

26. I understand that there are several rationales for combining prior-art references. For instance, I understand that it is considered obvious to: combine prior-art elements according to known methods to yield predictable results; substitute one known element for another to obtain predictable results; use the prior-art elements in a predictable way according to their established functions; apply a known technique to a known device (method or product) ready for improvement to yield predictable results; choose from a finite number of identified, predictable solutions, with a reasonable expectation of success; or if there is some teaching, suggestion, or motivation in the prior art that would have led a POSITA to modify a prior-art reference or combine prior-art teachings to arrive at the claimed invention. I understand, however, that a POSITA does not need to have the same motivations or

reasons to combine the prior art as the inventors, and can arrive at the claimed invention for completely different reasons than the inventors.

27. However, I also understand that the prior art cannot be modified or combined using “hindsight.” I understand that hindsight refers to situations in which the patent-at-issue is used as a framework to pick and choose pieces from the prior art to arrive at the claims, or adopting rationales identified by the patent-at-issue that those in the art would not have recognized. Instead, I understand that the obviousness inquiry is evaluated from the perspective of a POSITA as of the critical date of the patent-at-issue, with all the prior art before them, to determine whether the POSITA would have independently arrived at the claimed invention without the benefit of the patent-at-issue for guidance.

28. I understand that “objective indicia” or “secondary considerations” of non-obviousness include whether: (1) there was a long-felt need for the claimed invention; (2) the claimed invention has achieved commercial success; (3) there was copying of the claimed invention by others; (4) there were failed attempts by others to make the alleged invention; (5) there was praise of the invention in the field; and (6) there was skepticism the claimed invention could be achieved, among a few others.

29. As of the time that I prepared this declaration, the Patent Owner had not identified any secondary considerations of non-obviousness. Therefore, I reserve

the right to elaborate on my understanding of these secondary considerations and to address any secondary considerations that Patent Owner may subsequently identify.

30. Finally, I understand that, in an *inter partes* review proceeding, the obviousness of a claim must be demonstrated by “a preponderance of the evidence,” and that this burden falls on the Petitioner. I understand that the preponderance of the evidence standard requires that a reasonable factfinder could find a material fact is more probable than the nonexistence of that fact. It does not allow for speculation regarding specific facts and is instead focused on whether the evidence more likely than not demonstrates the existence or non-existence of specific material facts. I understand that “preponderance of the evidence” is a lower standard than “clear and convincing evidence” (which requires a fact to be substantially more likely to be true than untrue), or “beyond a reasonable doubt” (which is an exceedingly high standard that I understand is generally reserved for criminal matters).

31. I have applied the “preponderance of the evidence” standard throughout my declaration.

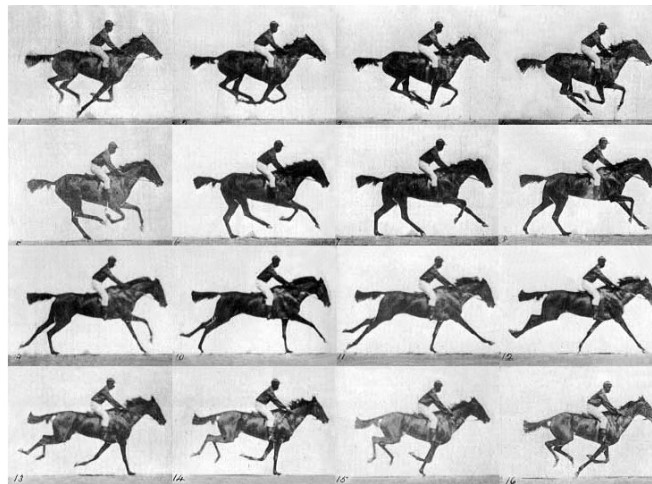
IV. TECHNOLOGY BACKGROUND

A. Video Compression

32. The ability to record a video, store it, transmit it from one location to another (whether by, e.g., broadcast television, satellite, cable lines, DVD or Blu-

ray disc, or the internet), and then play it back requires an extensive amount of behind-the-scenes processing that often goes unappreciated.

33. One of the problems is that raw videos can take up a significant amount of data. In its most basic form, a video is just a series of images—also called “pictures” or “frames”—that are played in rapid succession, with each image representing objects captured fractions of a second apart. For humans to perceive these separate images as being connected and in motion (e.g., as a video), some minimum number of images (e.g., 7 to 10 images) must be shown per second, and increasing the number of images per second increases the perceived smoothness of the motion. *See, e.g., Ex-1018.*



34. Although videos were originally recorded on film, with the rise of computers, videos are now stored digitally. To store and play back digital videos, the images that make up the video are first “encoded” from their image form into a bitstream—a stream of digital binary data often associated with the values of “zero”

and “one,” or “positive” and “negative” states. This step is necessary for digital computer processors and memory to store and play back digital videos, since these computer structures operate on digital data. To play back the video at a later time—after it has been stored in memory or transmitted from one device to another—the digital bitstream is processed and converted back (or “decoded”) into its image form.

35. However, unlike analog film that recorded every image in a video in full, encoding every image in a digital video would consume significant amounts of digital data. For instance, raw 4K video typically has a resolution of 3840 by 2160 pixels, or over 8 million pixels in each image that makes up the video. If the image uses 24-bit color (meaning the image can capture up to 2^{24} , or about 16.7 million, possible color values), then the size of each image would be approximately 25 megabytes using the following equation:

$$(3840 * 2160 \text{ pixels}) * \frac{24 \text{ bits/pixel}}{8 \text{ bits/byte}} * \frac{1 \text{ megabyte}}{1,000,000 \text{ bytes}}$$

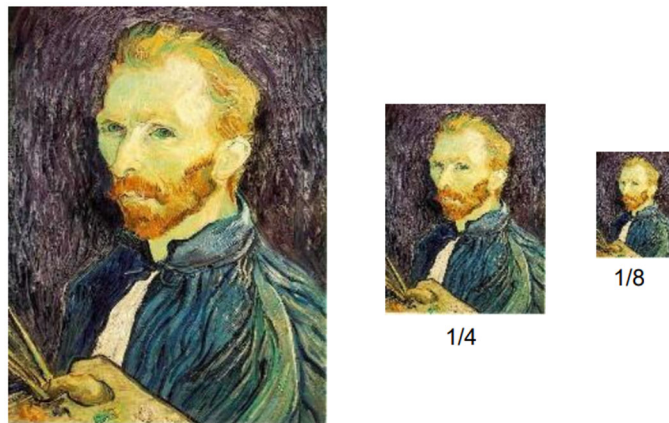
If each second of video had 30 images, then one minute of 4K video would require approximately 45 gigabytes of hard drive space. Storing and transmitting so much data would be impractical in most situations.

36. Accordingly, various ways of “compressing” digital video have been developed to both reduce the computational burdens and significantly reduce the file size of the videos.

1. Resampling/Resizing

37. First, because a video is just a series of images, each image in the video can be compressed just like any ordinary image would be. One easy way to do that is to reduce the size of the images. For instance, an HD image (as opposed to 4K) has one quarter of the number of pixels (approximately 2 million versus 8 million), and therefore requires only one quarter of the data to encode it in raw format (6 megabytes versus 25 megabytes).

38. There are several ways to reduce the size of an image, which is also called “downsampling” or “subsampling.” For instance, in its most basic form, the size of an image can be reduced by deleting every other row and column of pixels in the image. Ex-1019, 4.



Although simple, this form of downsampling can produce extremely “grainy” images with artifacts such as spatial aliasing, pixelization, and jagged edges when scaled for size to show the image on a particular display screen:



Id., 5. To produce better-quality downsampled images, rather than just deleting data, additional mathematics could be applied to effectively calculate a single, low-resolution pixel value based on multiple pixels in the original higher-resolution image:

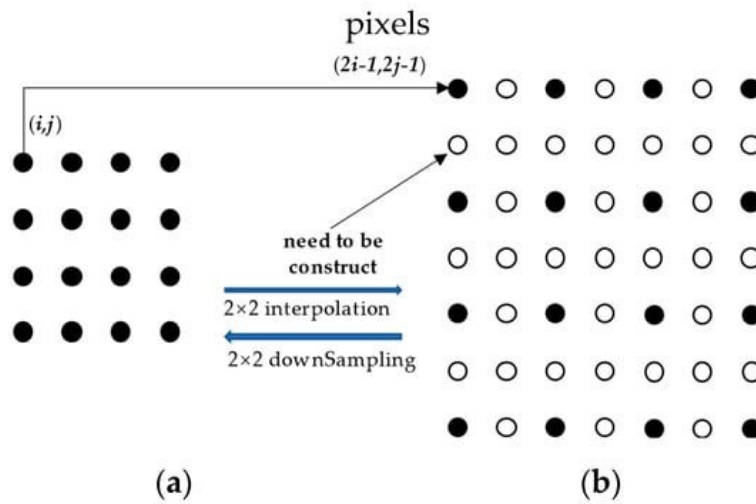


Id., 12. Although this often produces a blurrier image the more the image size is reduced, the resulting image is generally of higher quality than simply deleting data.

39. This process can also be approximately reversed by “upsampling” images to increase their effective resolution using a process called “interpolation.”¹ *Id.*, 19. Like downsampling, upsampling can use a variety of different mathematical techniques to essentially add pixels to the image, and estimate the values of those additional pixels that fall *between* those that already exist in the original or downsampled image. *Id.*, 20-24.

40. An example of this interpolation process is provided below, with the left side showing black pixels in a lower-resolution image, and the right side showing an upsampled version of the image that effectively doubles the resolution, where the pixels in white are interpolated (or estimated) based on the values of the black pixels in the original image:

¹ An “interpolation” process is not limited to upsampling, however. For instance, many in the field would consider a downsampling process that calculates a single pixel value in a new, lower-resolution image based on a weighted average of several pixels in a higher-resolution image, as shown in the figure above, to also be an “interpolation” process.

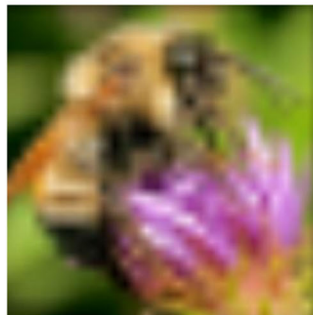


Ex-1064, 3. These white pixel values can be interpolated a number of different ways to produce different-quality results. Although the specific processes are not relevant to the '877 Patent, a few example interpolation methods with real-world results are provided below:

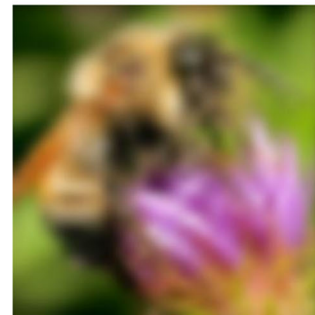
Original image:  x 10



Nearest-neighbor interpolation



Bilinear interpolation

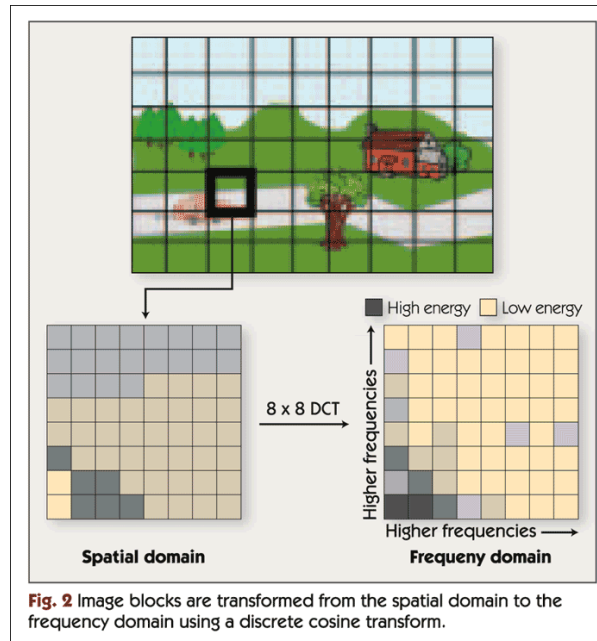


Bicubic interpolation

Ex-1019, 25.

2. Block Coding

41. Static images can also be compressed in ways other than just changing their size. For instance, another well-known way to compress an image is to split the image up into “blocks” and encode each block based on its contents.



Ex-1020, 2. In the example above, for instance, some blocks capture only the sky, which uses a few shades of blue. Rather than encoding the pixel values of a block directly, the pixel values can be transformed to the frequency domain by applying a transform such as the discrete cosine transform (“DCT”). Then, only the most significant (i.e., the largest) transform coefficients can be retained. Thus, blocks with minimal differences within them can be encoded using minimal data, and only blocks with finer details would require more data for encoding.

3. Predictive Coding

42. The previous examples have all focused on compressing *images*, but compressing video has even more tools at its disposal. In fact, one of the principal concepts guiding video compression is the notion of “prediction” *between* successive images in the video. Ex-1021, 19. Specifically, it was recognized nearly a century ago that “much of the depicted scene is essentially just repeated in picture after picture without any significant change, so video can be represented more efficiently by sending only the *changes* in the video scene rather than coding all regions repeatedly.” *Id.*, 19-20 (emphasis added). This is called “inter-picture” or “inter” coding. *See id.*, 20.

43. It was further recognized that “[m]ost changes in video content are typically due to the motion of objects in the depicted scene relative to the imaging plane.” *Id.*, 20. For instance, if a video captures a car driving down a road, most of the objects in each image of the video will stay the same, and some may just move relative to one another (such as the car relative to the road in the example below). Rather than encode every image from scratch, the *difference* between each image in the video can be encoded: in the example below, the motion of the car.

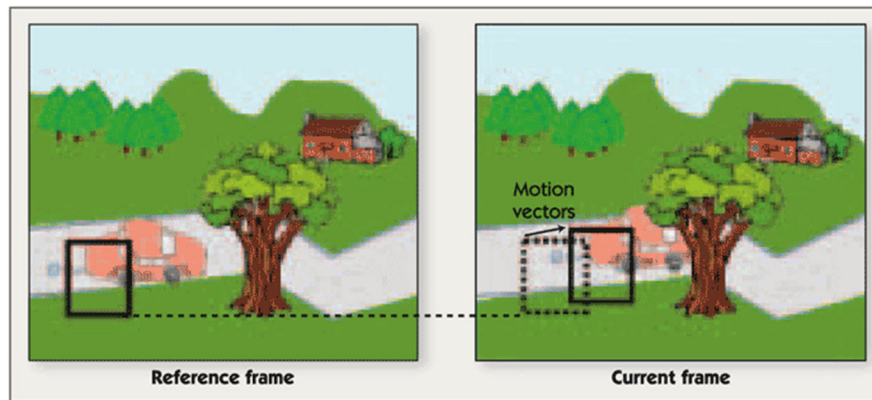


Fig. 4 Motion estimation predicts the contents of each macroblock based on motion relative to a reference frame. The reference frame is searched to find the 16x16 block that matches the macroblock, motion vectors are encoded, and the difference between predicted and actual macroblock pixels is encoded in the current frame.

Ex-1020, 6.

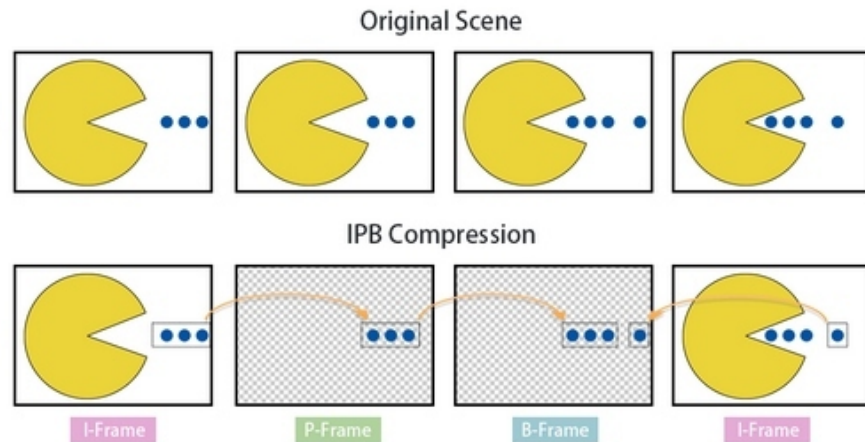
44. To encode the differences between images, an encoder will typically break the current image up into a series of blocks and search a “reference” frame—which is just another image in the video (either a previous *or* future frame)—to determine whether the current block matches a portion of the reference frame. When a match is found, the encoder will calculate a “motion vector” that characterizes how the objects in the block have moved between the current frame and reference frame. Ex-1020, 5-6; Ex-1021, 20. In the example above, for instance, the current block (on the right) covers the rear of the car, and so the encoder searches for a match to that block in a reference frame (on the left), and then characterizes how that portion of the car moved between frames. Ex-1020, 5-6; Ex-1021, 20. Rather than encode/decode the contents of the block for the current frame, the current frame can

simply take the reference frame and modify it slightly through “motion compensation” using the motion vectors. Ex-1020, 5-8; Ex-1021, 20.

45. However, because motion compensation cannot account for *every* difference between a current frame and reference frame (e.g., imagine the car’s wheel rotated a bit relative to the reference as it moved down the road), after motion compensation, a “residual” can be calculated that captures any remaining differences between the reference frame and current block. Ex-1021, 20; Ex-1020.

46. In sum, predictive video compression relies upon a few different types of frames. Specifically, frames that are compressed based only on their own contents, and do not need to refer to other reference frames, are called “I” frames (or “intra-” or “key-frames”). Ex-1020; Ex-1011 (“Fimoff”), 1:43-47. In other words, I frames in a video are compressed like any ordinary standalone picture would be, such as using block-based encoding that I had described previously for still images. The I frames can then be reconstructed without relying on any other frames in the video. Frames having blocks that are compressed using *predictions* from a single reference frame are called “P” frames (or “predicted” frames). Ex-1020; Fimoff, 1:43-47. And frames having blocks that are each compressed using *predictions* from up to two reference frames (in MPEG-2, for example, at least one past reference frame and at least one future reference frame) are called “B” frames (or “bi-

directional” frames). Ex-1020; Fimoff, 1:43-47. An example of I, P, and B frames, as well as their relationships, is provided below:



Ex-1065, 2.

47. Every video that uses blocks from one or more reference frames to perform predictive compression (also known as “inter-frame” or “inter” prediction) therefore needs to be coded using at least one I frame to serve as the starting point for any subsequent P and B frames. However, when the contents of a video change significantly, such as during a camera cut or a change of scene, a new I frame is typically required to then serve as the basis for subsequent P and B frames.

48. An example predictive encoder is provided below, which generally follows a conventional predictive-coding process:

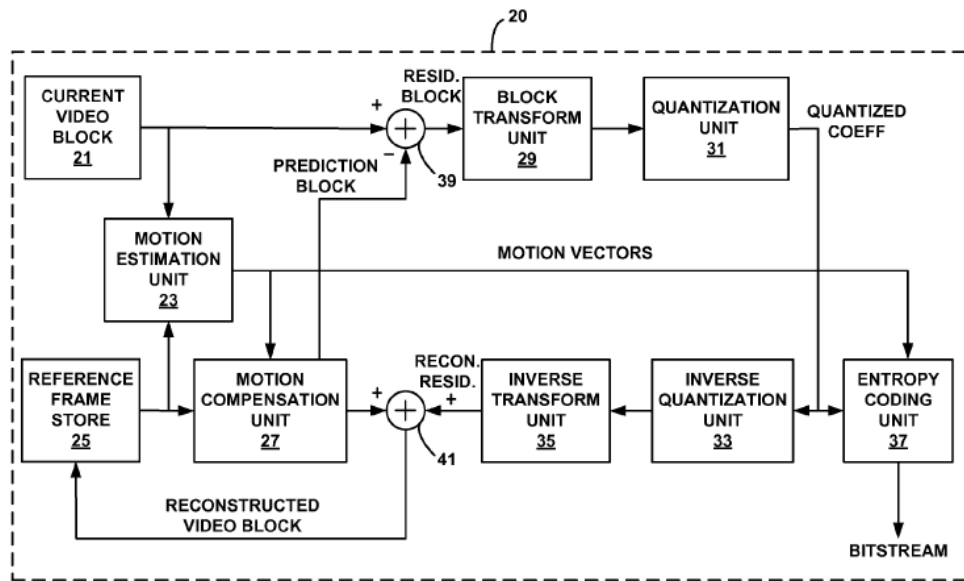


FIG. 2

Ex-1014, Fig. 2. First, the encoder receives a block (21) from the current video image. *Id.*, ¶54. If the block is from an I frame, that means it is not being predicted based off any reference frames, and it simply passes through the “block transform unit” (29) and “quantization unit” (31) to apply block-based compression, as discussed above, before being sent to an “entropy coding unit” (37) to create the final, encoded bitstream that is either transmitted or stored for later decoding. *Id.*

49. However, because this is a *predictive* encoder, that does not end the process. Instead, the block-coded I-frame (the output of the “quantization unit” (31)) is sent not only to the entropy coding unit (37), but is also sent through what is called a “decoding loop,” which is effectively a decoder built into the encoder. I have annotated the components of the decoding loop in red in the block diagram below:

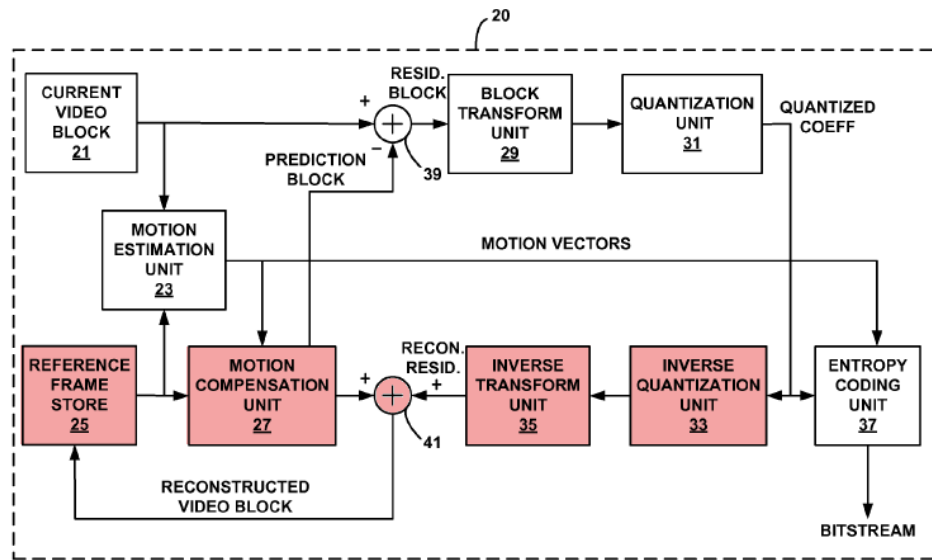


FIG. 2

Ex-1014, Fig. 2 (colors added). Predictive encoders typically include a decoding loop to generate the reference frames from which P and B frames are predicted. The encoders use reconstructed versions of images—rather than the original images (e.g., 21)—as reference frames because reconstructed images differ slightly from the original due to the compression processes applied during encoding. *See, e.g.,* Ex-1007 (“Nakagawa”), 5:47-53. And because the decoder will only have access to reconstructed versions of the reference images—and not the original versions of the reference images—encoders must also generate reconstructed versions of the images so that any predictive parameters (i.e., motion vectors and residuals) calculated by the encoder will match the version of the images they are being applied to at the decoder. *See, e.g.,* Ex-1033, 73 (“It is very important that both encoder and decoder use identical reference frames to create the prediction. Otherwise, the predictions in

encoder and decoder may be different, leading to an increased error between [the two].”).

50. When the signal enters the decoding loop, the quantization (31) and transform (29) processes are first undone by “inverse quantization unit” (33) and “inverse transform unit” (35). For I frames, those may be the only operations performed.² The resulting reconstructed I frame is then stored in “reference frame store” (25). *See* Ex-1014, ¶58.

51. After an I frame is encoded, reconstructed, and stored in the reference frame store (25), it may then serve as the basis to predict subsequent P and B frames. When the “current video block” to be encoded comes from a P or B frame, the block will be sent to a “motion estimation unit” (23) that “compares video block 21 to blocks in one or more adjacent video frames to generate one or more motion vectors” by “identif[ying] a block in an adjacent frame that most closely matches the current

² In the context of MPEG-2, for example, those are generally the only operations performed. But later video coding standards such as H.264 utilize sophisticated intra prediction methods where the pixels of a block in an I-frame may be spatially predicted from the values of other pixels *in the same frame* (i.e., without using any reference frame). However, the details of such intra prediction processes are not central to the alleged novelty of the ’877 patent. *See, e.g.*, Ex-1001, 10:11-14.

video block 21 ... and determin[ing] a displacement between the blocks. On this basis, motion estimation unit 23 produces a motion vector that indicates the magnitude and trajectory of the displacement.” *Id.*, ¶55.

52. After the “motion estimation unit” (23) determines the appropriate motion vector, the vector is provided to both the entropy coding unit (37) for encoding into a bitstream and, separately, a “motion compensation unit” (27) in the decoder loop. As part of the decoding loop, the motion compensation unit (27) applies the motion vector to the reference block to calculate a “prediction video block.” *Id.*, ¶57. This “prediction video block” is then subtracted (39) from the “current video block” (21) to produce the residual signal, which represents the remaining differences between the motion-compensated prediction block and the original block. This residual is then sent through the block transform (29) and quantization unit (31) just like in the case of an I frame. And just like the I frame, not only is this transformed/quantized residual sent to the entropy coding unit (37) to be encoded into the bitstream, but it is also sent back into the decoding loop where it too is inverse quantized and inverse transformed, and added back to the “prediction video block” to create the reconstructed image that is then stored in reference memory. *Id.*, ¶58. In sum, for P and B frames, the predictive encoder will take an input video signal and produce a bitstream comprising motion vectors and a residual signal (among other control signals) for storage or transmission out of the encoder,

as well as a reconstructed version of the P or B frame to serve as a reference for subsequent frames.

53. A typical decoder will then have many of the same components as the encoder's decoding loop. An example of a predictive decoder is shown below, which has an "inverse quantization module" (34), "inverse transform" (36), "motion compensation module" (30)" and "reference frame store" (40) like the decoding loop in the encoder:

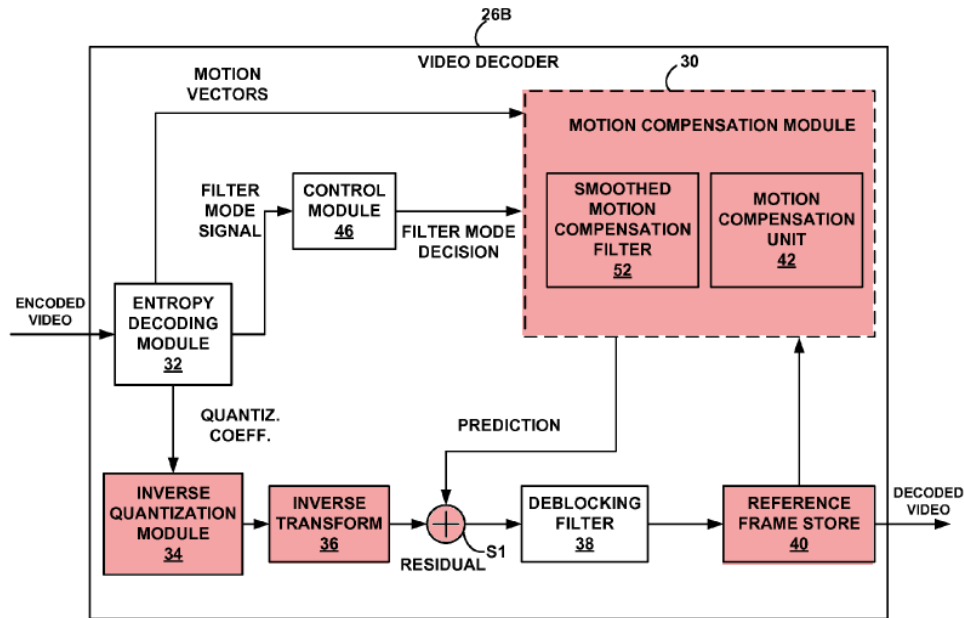


FIG. 5

Ex-1014, Fig. 5 (colors added). This example decoder also has a few additional modules, such as a "control module" (46) and "deblocking filter" (38) which performs additional operations to assist with the decoding process. At base, though, the decoder follows a similar process as the encoder's decoding loop: it receives the

motion vectors and residual from the encoder, applies the motion vectors to a reference block in reference frame store (40), and adds the motion-compensated block to the inverse quantized/transformed residual (and performs some additional filtering) to produce the decoded video.

4. Combining Compression Methods

54. Over time, those in the art had experimented with many different ways to compress video, including combining different aspects of the operations described above, as well as many other more-complex compression solutions built upon these basic principles. For instance, the use of predictive compression (and I, P, and B frames) has been widely adopted in video compression standards since the 1990s. The rise of streaming video over the internet, however, has demanded further development in video-compression methods. Specifically, when digital videos were first transmitted using physical media (e.g., DVDs), the videos would need to be encoded once (when written to the DVD), physically transmitted (e.g., shipped) to the intended viewer, and could then be decoded by any compatible media player (e.g., any DVD player). However, when digital videos are transmitted in real time over the internet instead of in fixed, physical media, whether a video can be transmitted from a host's servers to the viewer's device depends on the availability of sufficient network bandwidth. If networks are congested (e.g., because the network is already occupied transmitting different data), for instance, the video

transmission must either adapt to those network conditions and work within the available bandwidth or risk transmission delays that could negatively impact playback.

55. For this reason, in the late 2000s, those in the art had also begun experimenting with “adaptive” resolution encoding, Ex-1022, 1, which combined predictive compression with resizing or other operations. For instance, a host may store digital videos at multiple different resolutions (or sizes) and change which resolution of video is sent over the network in real time as network conditions change. *Id.*, 3-4. Alternatively, “scalable video coding” could be employed, in which case a “base layer” (typically, low-quality video) could be sent over the network, and then further “enhancement layers” could be added to improve the quality of the video as more bandwidth became available. *Id.*, 4. In sum, though, these “adaptive” solutions combined existing predictive coding techniques with various resizing or enhancement operations to account for network-capacity issues.

B. Standards Development

56. Because encoding and decoding video requires converting images into digital data and then reconstructing the images based on that data, it was recognized early on that video compression processes should be standardized to ensure videos could be decoded by a wide variety of devices.

57. In the late 1980s and early 1990s, for instance, two bodies had developed their own standards for video encoding and decoding: the International Telecommunication Union (ITU)—an agency of the United Nations—had developed the H.261 standard; and the Moving Picture Experts Group (MPEG)—a joint venture of the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC)—had developed the rival MPEG-1 standard.³ Ex-1023. However, in the mid-1990’s the ITU and MPEG groups joined together to build the MPEG-2/H.262 standard and continued releasing improved standards such as Part 10 of MPEG-4/H.264 (also known as “Advanced Video Coding” or “AVC”) in the late 1990s. *Id.*, 2-3.

58. While the early 2000s remained relatively quiet as ITU and MPEG iteratively updated MPEG-4/H.264, in 2010, ITU/MPEG announced that a new team was being formed “to develop a new generation video coding standard that will further reduce by 50% the data rate needed for high quality video coding, as compared to the then state-of-the-art AVC standard.” Ex-1030. This new standard would be called the “High Efficiency Video Coding (HEVC)” standard. *Id.* The team

³ Private companies were also developing their own video codecs, such as Microsoft’s AVI and WMV codecs.

in charge of developing the HEVC standard was known as the “Joint Collaborative Team on Video Coding” (“JCT-VC”). *Id.*

59. HEVC is also known as H.265, the successor to MPEG-4/H.264.

V. '877 PATENT

60. The '877 Patent was filed on July 13, 2014. It was filed as a national-stage entry of international patent application PCT/EP2013/050399, which was filed on January 10, 2013, and claims priority to an even earlier French application (No. 12 50334) filed on January 13, 2012.

61. I therefore understand that for purposes of comparing the '877 Patent to the prior art, the earliest possible priority date is January 13, 2012.

62. I have reviewed the '877 Patent and provide the following summary.

A. Specification

63. The '877 Patent is primarily concerned with addressing a situation that arises when existing adaptive resolution and predictive compression methods are combined. That is, when an encoder adaptively changes the resolution of the video in response to network conditions, a situation may arise in which the decoder will need to predict a current frame based on a different-sized reference frame. '877 Patent, 1:51-55. The '877 Patent depicts an example of this situation in Figure 3:

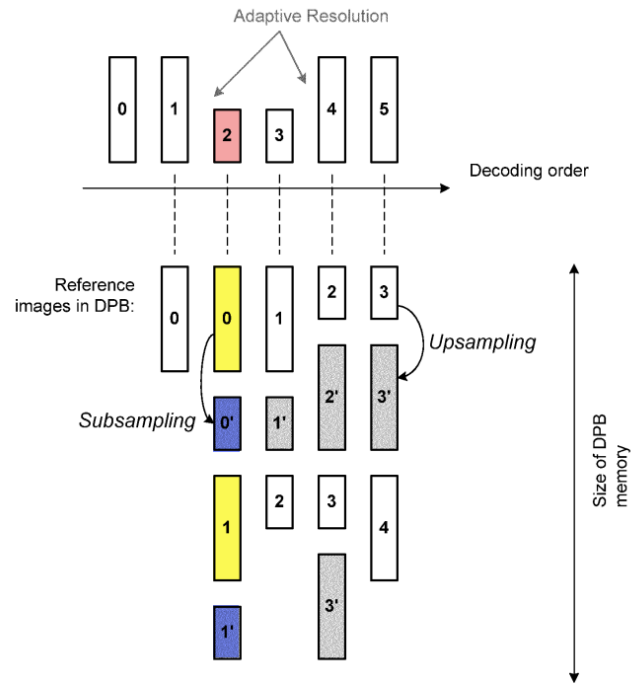


FIGURE 3

64. The blocks above the horizontal arrow represent the sizes of five different frames being received by the decoder. Specifically, two frames (2 and 3) were encoded and transmitted at a smaller size or resolution compared to the other frames (0, 1, 4, and 5).

65. The blocks below the horizontal arrow represent (already-reconstructed) frames that are being used as references to predict each of the above-line frames. In other words, frame 1 is a P frame that is being predicted using (the previously reconstructed) frame 0 as a reference, frame 2 is being predicted using frames 0 and 1 as references, and so on. For each received frame shown above the

horizontal arrow, the reference frames being used for prediction are shown in a column below the horizontal arrow.

66. Figure 3 captures the situation that can arise due to adaptively resizing frames in a video stream: sometimes the size of the reference frames will not match the size of the current frame being decoded. I have annotated one example in Figure 3, which shows that the current frame 2 (in red) is being predicted using larger frames 0 and 1 (in yellow).

67. To accurately predict a frame, however, the size of the reference frame should match that of the current frame. *Id.*, 1:55-59. Otherwise, the motion vectors, predictions, and residuals calculated by the encoder will not line up properly when applied to a case where the current frame and reference frame have different sizes. To make them the same size, the decoder must generally resample (either “upsample” or “downsample”) the reference frames to match the size of the current frame being predicted. *Id.* In this example, reference frames 0 and 1 would be resampled to match the size of frame 2, producing re-sized reference frames 0’ and 1’ (shown in blue in annotated Figure 3, above). Re-sized frames 0’ and 1’ would then be used as references to predict frame 2.

68. According to the ’877 Patent, however, these resampled reference frames (e.g., those shown in blue in annotated Figure 3, above) would need be stored in a “decoded picture buffer” (“DPB”) along with the original-sized reference

frames. *Id.*, 1:59-61. But the memory used to store all of these reference frames (also called “buffer” memory) was known to be a limited resource and increasing the required memory size to allow for storing reference frames at multiple resolutions was a known problem for hardware implementations of encoders and decoders. *Id.*, 2:8-13.

69. For this reason, the '877 Patent proposed combining the resizing (i.e., resampling) and prediction (e.g., motion compensation) operations, *id.*, 2:32-37, so that current frames could be predicted using different-sized reference frames without having to store the resampled reference frames in the buffer, *id.*, 6:1-7. Instead, “only the reconstructed reference images [would be] stored in the [decoded picture buffer] memory, i.e. the images not greyed out in FIG. 3.” *Id.*, 5:48-52.

70. The '877 Patent also restated what by then had already been well-understood: that a filter (here, a two-dimensional motion-compensation, upsampling, and subsampling filters) that can perform filtering in two directions in a single operation “are generally separable filters,” and thus a separate vertical and horizontal filters could be applied respectively to each column of pixels and each row of pixels individually for implementing filtering operations such as resampling and motion compensation.. *Id.*, 4:28-36. Applying a filter as either a single, two-dimensional filter or two directional, separable filters was known in the field. *See, e.g.,* EX-1015, 6:14-26 (“[A]pplying ... a vertical one-dimensional filter and [a]

horizontal one-dimensional filter to the same region ... provides a two-dimensional filter effect, however, with a reduced complexity compared to an application of a two-dimensional filter.”).

B. Prosecution History

71. I have reviewed the '877 prosecution history and provide the following summary.

72. As I mentioned above, the '877 Patent is a national stage entry of an earlier-filed international application. Ex-1002, 52. After filing the U.S. version of the application, the claims were amended before any examination had begun, resulting in twelve claims being originally presented for examination. *Id.*, 5-9. These included independent claims to decoding and encoding methods (claims 1, 6) and decoding and encoding devices (claims 11, 13). *Id.*

73. After the first round of examination, the Examiner rejected all of the pending claims as obvious in view three prior-art references: Ye (U.S. Patent No. 8,199,812), Hong (U.S. Application No. 2013/0003847), and Karczewicz (U.S. Application No. 2009/0175336). *Id.*, 167-72.

74. In response, the Applicant amended the claims to recite that the two claimed filtering operations—the resampling and motion-compensation-filtering operations—are each performed by a single vertical and single horizontal filter. *Id.*, 189-94. The Applicant argued none of the prior art disclosed resampling and motion

compensation being performed by a single filter in each direction because the prior art used different modules for motion compensation and resampling. *Id.*, 196-200. The Examiner, however, maintained the prior-art rejection. *Id.*, 213-19.

75. The Applicant then amended the claims again to specify that reference images would be stored in a “decoded picture buffer,” but resampled versions of those reference images (to change their size) could *not* be stored in the buffer. *Id.*, 227-34. And although the claims previously described the resampling and motion compensation filters using words, the Applicant rewrote their description to claim them in equation form. *Id.* The Applicant then argued that these new features distinguished the prior art relied upon by the examiner. *Id.*, 235-38.

76. The Examiner still did not allow the claims, though, and issued another rejection using a new combination of prior art: Ye, Tsai (U.S. Application No. 2012/0082241), Robertson (U.S. Application No. 2010/0226437), and Sato (U.S. Patent No. 6,748,018). *Id.*, 252-58.

77. This time, the Applicant amended the claims to specify that the resampling and motion-compensation filters were “applied jointly,” *id.*, 274-79, and argued that the cited prior art taught only “horizontal resampling *followed by* motion compensation,” *id.*, 280-82.

78. After this last amendment, the Examiner allowed the claims. The Examiner concluded that “[t]he claimed invention now relates to more uniformly

[sic] application jointly of either vertical and horizontal interpolation filtering and resampling to respective single filters in either vertical or horizontal applications.” *Id.*, 294-97.

VI. PRIOR ART

A. Nakagawa (Ex-1007)

79. Nakagawa is a U.S. patent that was filed on November 4, 1996, and issued on September 8, 1998. Ex-1007, Cover. I understand Nakagawa claims priority to an earlier Japanese application filed March 29, 1996.

80. Because Nakagawa issued (September 8, 1998) more than a year before the earliest possible priority date of the '877 Patent (January 13, 2012), I understand that Nakagawa constitutes prior art under at least pre-AIA § 102(b).

81. After identifying the major video codec standards at the time (H.261, MPEG-1, and MPEG-2) and providing a high-level summary of their operation, Nakagawa, 1:10-20, Nakagawa explains that an issue may arise “[w]hen a motion of considerable magnitude or a full scene transition happen[s] in the middle of a sequence of video frames,” *id.*, 1:21-23. In those instances, an encoder “may suffer from an overwhelming amount of coded frame data that exceeds a standard level allowed for each frame,” which causes the encoder to “forcibly reduce the amount of coded data in an attempt to regulate it at the standard level.” *Id.*, 1:23-26. According to Nakagawa, this process “will cause extreme degradation of image

quality and/or too coarse frame sub-sampling ... resulting in unacceptably poor pictures when reconstructed at receiving ends.” *Id.*, 1:25-31. Nakagawa then explains that a prior art system proposed to solve this problem based on estimation of the average motion vector magnitude is inadequate – at least because it may “needlessly lower the resolution” for multiple reasons including “indifference to buffer occupancy” and because it introduces new data dependencies that are detrimental to pipelining. *Id.*, 1:52-2:46.

82. According to Nakagawa, one way to avoid this problem is to alter the size of the encoded images in real time to “regulate the amount of coded data at an appropriate level.” *Id.*, 2:49-53. Although at least one other prior-art system had tried to implement this solution, Nakagawa considers it insufficient because the factors used to determine whether the size of an encoded image needed to be changed could apply a resolution change unnecessarily, and would not be able to make the determination fast enough. *See id.*, 1:32-2:46.

83. Nakagawa proposes an alternative solution. Specifically, Nakagawa uses the H.261 standard as an example and explains how, under that standard, video images can be encoded to be of one of two sizes (which are called “CIF” or “QCIF”). *Id.*, 4:43-51. After each frame is encoded at a particular size, Nakagawa’s system evaluates several factors about the most recently encoded frame—including at least the “quantizer step size, the amount of coded data, and the buffer occupancy,” *id.*,

3:35-38—to determine whether the size of the next frame to be encoded should be changed. *See, e.g., id.*, 3:64-4:6.

84. However, because the size of each encoded frame in a video sequence may change with Nakagawa’s system, this means that sometimes a current frame will be reconstructed using predictions from a different-sized reference frame. *Id.*, 6:15-40. To solve that problem, Nakagawa also provides a “CIF/QCIF converter” that “retrieves the reconstructed picture of the previous frame from the frame memory 22 and converts it” to match the size of the frame being predicted, *id.*, before performing a routine reconstruction operation needed to encode the current frame, including applying motion vectors to the (resized) reference frame so that the encoder can calculate the residual for the current frame. *See id.*, 4:52-5:20.

B. Sita (Ex-1008)

85. Sita is a U.S. patent application publication that was filed on September 26, 2003, and published on August 5, 2004. Ex-1008, (“Sita”).

86. Because Sita was published (August 5, 2004) more than a year before the earliest possible priority date of the ’877 Patent (January 13, 2012), I understand that Sita constitutes prior art under at least pre-AIA § 102(b).

87. Sita discloses a decoder for MPEG-2 videos, which Sita recognizes is compatible with many different types of video formats of various sizes. *See Sita*, ¶¶2-4, 66-67. Sita explains, however, that there is a need for a single decoder that is

compatible with all of these different formats, and that also supports the conversion of high-definition content to be compatible with standard-definition displays. *Id.*, ¶7.

88. On this basis, Sita discloses a MPEG-2 compatible decoder that can receive, e.g., high-definition signals that are ultimately downsampled to reduce the size of the video image after it has been decoded and reconstructed. *See id.*, ¶¶67, 82. Sita not only outputs these downsampled images to standard-definition displays, but also stores these downsampled versions of the reconstructed images in memory to serve as reference frames to predict subsequent images. *Id.*, ¶82. According to Sita, storing only downsampled versions of reconstructed images “result[s] in a considerable reduction of memory required for storing reference images.” *Id.*, ¶83.

89. However, because Sita’s decoder will continue receiving high-definition signals—including high-resolution motion vectors and residuals—from the encoder, but all reference images are stored at standard definition, there would be a mismatch between the size of the image being reconstructed and the image being used as a reference frame. To account for this mismatch, Sita explains how the decoder will apply a combined resampling/motion-compensation operation to convert the standard-definition reference images stored in memory back into high definition. *Id.*, ¶¶88, 89, 110-112, Fig. 2B. That way, the high-definition resolution motion vectors from the encoder can be applied to reconstruct a prediction of the current image. *Id.*

90. Although Sita describes in detail resampling and motion compensating frames in the horizontal direction, Sita also recognizes that “[o]ne skilled in the art could easily extend [the teachings about the horizontal filter] to motion translation in the vertical direction if desired.” *See, e.g., id.*, ¶95.

C. Wiegand (Ex-1009)

91. Wiegand is a working-draft proposal for the High-Efficiency Video Coding standard, which was published on June 27, 2011. *See* Ex-1045 (version 8). It was published by the JCT-VC working group responsible for drafting the HEVC standard. The version Wiegand published was a product of the JCT-VC group’s fifth, March 2011 meeting. Ex-1009 (“Wiegand”), 1.

92. Because Wiegand was published (June 27, 2011) before the earliest possible priority date of the ’877 Patent (January 13, 2012), I understand that Wiegand constitutes prior art under at least pre-AIA § 102(a).

93. Wiegand explains that the purpose of the JCT-VC’s efforts is to propose an “industry standard for compressed video representation with substantially increased coding efficiency and enhanced robustness to network environments.” Wiegand, 12 (Forward), 13 (Prologue §0.1).

94. As part of the draft standard, Wiegand discloses a process for decoding bitstreams of video data. *See generally*, Wiegand, 84-154 (§8). Specifically, Wiegand discloses that a decoder receives a bitstream from the encoder (in the form

of “NAL units”), *id.*, 85 (§8.1), which is then decoded to obtain reconstructed video frames that may have been designated by the encoder to serve as reference frames, *id.* (§8.2.2); *id.*, 90-91 (§8.2.3). When Wiegand’s decoder is in an “inter prediction mode” in which an incoming block is set to be predicted and reconstructed based on a reference frame, Wiegand describes performing a motion compensation operation in which the prediction block selected by the encoder is located by the decoder using a motion vector signaled (either explicitly or implicitly) in the encoded bitstream by the encoder for this purpose. *Id.*, 109 (step 1, referencing §8.4.1), 110-11 (§8.4.1 instructing to follow §8.4.2), 111-30 (describing derivation and interpolation of motion vectors). Then, the decoder decodes the residual signal that must be added to the prediction block in order to reconstruct the decoded version of the current block and ultimately to reconstruct the entire decoded image, *id.*, 109 (steps 2-4, referencing §8.4.3), 130-33 (§8.4.3).

D. Davies (Ex-1010)

95. Davies is a proposal that also arose from the JCT-VC working group for the HEVC standard. Davies was published on July 7, 2011, Ex-1050 (version

4),⁴ so that it could be considered at the JCT-VC's sixth meeting in July 2011. It was published just over one week after Wiegand.

96. Because Davies was published (July 7, 2011) before the earliest possible priority date of the '877 Patent (January 13, 2012), I understand that Davies constitutes prior art under at least pre-AIA § 102(a).

97. Davies expressly proposes modifying Wiegand. *See* Ex-1010 ("Davies"), 4 (§ 4) ("Various modifications to the current Working Draft (WD) [3] are needed"), 11 (citing Wiegand as reference "[3]"). In particular, Davies proposes a process for "resolution switching" "to allow video communications to use re-scaling to adapt seamlessly to adverse network conditions." *Id.*, 1. Davies notes how in AVC—otherwise known as the MPEG-4 Part 10/H.264 standard—"prediction across resolutions is not permitted," and instead a bandwidth-intensive "IDR frame must be sent to re-initialize the stream" when the resolution of a video is changed.

⁴ Although a package of documents was also published as "version 5" on the JCT-VC website on July 15, 2011, Ex-1050, the "version 5" package contains the same "version 4" proposal document that I call "Davies" here. The difference between the two is that "version 5" contains an additional, separate slide deck that was not included in the "version 4" package that also includes Davies.

Id., 2 (noting that IDR frames of sufficient quality are “difficult to transmit in a low-delay environment”).

98. Because Wiegand’s draft standard did not propose significant optimizations for IDR frames compared to those in AVC, but had much more success in optimizing inter-prediction encoding compared to AVC, Davies proposes “that an encoder can signal a change of resolution and predict frames across resolutions” without needing to send an IDR frame, and instead “reference frames are simply re-scaled as needed.” *Id.*, 2 (§1). To accomplish this on-demand rescaling, Davies proposes using “separable down- and up-scaling filters,” *Id.*, 4 (§3.2), which can be applied during the inter-prediction process disclosed by Wiegand, *id.*, 7 (§4.4.1) (suggesting changes to § 8.4.2.1.6 and § 8.4.2.2.1 of Wiegand). Thus, videos could be rescaled while operating in Wiegand’s inter-prediction mode without having to send the bandwidth-intensive IDR frames to reset the resolution of the video stream.

E. Fimoff (Ex-1011)

99. Fimoff is a U.S. patent that was filed on December 15, 1999, and issued on September 30, 2003. Ex-1011.

100. Because Fimoff was published (September 30, 2003) over a year before the earliest possible priority date of the ’877 Patent (January 13, 2012), I understand that Fimoff constitutes prior art under at least pre-AIA § 102(b).

101. Before summarizing Fimoff's disclosure, it is worth considering the context in which Fimoff came about. Around the time Fimoff was published, television networks were switching from broadcasting standard-definition content to high-definition content as high-definition televisions became more prevalent. Ex-1034. The Telecommunications Act of 1996 had also mandated a transition from analog to ATSC digital broadcast television in the United States. While the initial deadline for stations to switch to digital television broadcasting was December 31, 2006, the federally mandated deadline for full-power stations to end analog broadcasting was eventually pushed back to June 2009. However, because many people still had standard-definition televisions during the transition period, there was a desire to allow older, standard-definition televisions to play back high-definition content the networks had begun broadcasting. That was a problem Fimoff was trying to solve. Fimoff, 3:37-47.

102. At the time of Fimoff, the MPEG-2 video standard predominated. MPEG-2 was the precursor to MPEG-4 and HEVC. As Fimoff recognizes, MPEG-2 in part enabled the switch from standard-definition to high-definition television streams, because MPEG-2 allowed for "the encoding of video over a wide range of resolutions, including higher resolutions commonly known as HDTV." *Id.*, 1:25-28. Fimoff therefore begins by providing a high-level overview of the MPEG-2 encoding and decoding process. *Id.*, 1:48-3:36.

103. Fimoff then explains how a decoder receiving high-definition signals could be configured to output standard-definition content through a downsampling process. *Id.*, 1:15-18, 3:37-47. Although Fimoff focuses mostly on decoding P and B frames, a POSITA would have understood that Fimoff's decoder would first receive a high-definition I frame (because the I frame would be expected to be received first), which is downsampled during reconstruction (using the downsampling module 102) to produce a standard-definition image as its output. *See, e.g., id.*, 1:39-47 (describing treatment of I frames in MPEG-2 generally), 2:66-3:3; *see also id.*, 7:66-8:3, Fig. 5 (showing a downsampling module 102 that operates on the incoming signal). This standard-definition I frame would then be stored as a reference image for subsequent P and B frames. *Id.*, 3:1-3.

104. When a subsequent P or B frame would need to be predicted based on a past frame, however, an issue would arise. Specifically, because the encoder is encoding P and B frames using high-definition reference images, and sending high-definition frames for decoding, the decoder's standard-definition reference images would be smaller (e.g., have a lower resolution) than both the current high-definition frame entering the decoder *and* the reference image used by the encoder to calculate the motion vectors and residual (because the encoder was not downsampling its reference images to standard definition like Fimoff's decoder). *See, e.g., id.*, 4:5-15, 19:21-27. That means there would be a mismatch between the reference images used

by the encoder—which are used to calculate the motion vectors and residuals—and those used by the decoder—which applies the motion vectors and residuals calculated by the encoder; the decoder would be operating on a different-sized reference frame.

105. To account for this mismatch and ensure current video frames are predicted at the decoder using a reference frame of the same size as the encoder, Fimoff explains that the decoder’s standard-definition reference images are pulled from memory and upsampled to effectively convert the standard-definition reference images back into high-definition. *Id.*, 18:6-19. That way, the full-resolution motion vectors provided by the encoder (which were calculated using a high-definition reference image) can be accurately matched to the now high-definition reference image at Fimoff’s decoder. *See id.*, 19:22-31, Fig. 7.

106. After applying the high-definition motion vectors to the upsampled reference image to produce a high-definition predicted image, the predicted image is then downsampled again to get the image back to standard-definition. *Id.*, 20:25-36, 20:60-21:14, 21:52-55. Fimoff explains that, through this process, “prediction upsampling, linear interpolation [for motion compensation], and prediction downsampling,” can occur in “a single operation.” *Id.*, 21:15-17, 21:55-60, 23:60-24:5, 24:43-55.

107. As Fimoff explains, filtering occurs in both the horizontal (rows of pixels) and vertical (columns of pixels) direction. *Id.*, 30:20-38, Fig. 5. The output of the combined upsampling/interpolation/downsampling operation is a standard-definition prediction of the current image, which is then combined with a separately downsampled residual to produce the final reconstruction of the image in standard definition. *Id.*, 30:34-38. This reconstructed, standard-definition image is then used to display the (previously) high-definition content on a standard-definition display and is also stored in memory as a reference frame for subsequent images. *See, e.g., id.*, Fig. 5.

108. Also worth noting is that at the time of Fimoff, television broadcasts often transmitted “interlaced” video. *See* Fimoff, 3:4-27. “Interlaced” refers to an older video processing format in which each video frame is divided into two fields: one containing the odd numbered lines and the other containing the even numbered lines. When the two fields are displayed in rapid succession, the even and odd lines appear interwoven. An example of an interlaced image is provided below:



Field 1 - Odd Lines



Field 2 - Even Lines



Frame

Ex-1066. Interlacing is typically used in broadcast television to double the effective frame rate of the video (how many images per second are shown on the screen) without needing any additional bandwidth for each frame. Ex-1024. Each (partial) image in an interlaced video is referred to as a “field.” *See* Ex-1066. This is different than the more-modern “progressive” video format, in which entire pictures are displayed one at a time (as opposed to alternating odd and even lines of pixels). Ex-1024. Each image in a progressive video is referred to as a “frame.” *Id.*

109. To accommodate the encoding/decoding of interlaced video, Fimoff therefore focused on describing how its vertical filters operated because the vertical filters would need to deal with the issue of missing pixels in each field of interlaced video. Fimoff therefore described three particular arrangements for the vertical filter: (1) filtering images of all field blocks (a block with alternating rows of pixels available), *see* Fimoff, 8:57-9:55, 19:20-21:67; (2) filtering images of all frame blocks (a block with all rows of pixels available), *see* Fimoff, 10:32-13:25, 24:55-30:8; and (3) filtering images of mixed types of both, or between one and the other of, field and frame blocks, *see* Fimoff, 13:26-14:55, 22:1-24:54.

VII. LEVEL OF ORDINARY SKILL

110. I understand that when both interpreting the claims and assessing the prior art, I must do so from the perspective of a person of ordinary skill in the art

(“POSITA”) as of the relevant priority date. My understanding is that the earliest possible priority date of the ’877 Patent is January 13, 2012.

111. In my opinion, in the 2012 timeframe, a POSITA would have had at least a bachelor’s degree in electrical engineering, computer science, computer engineering, or a related field, and approximately three years of experience in design, development, and/or testing of relevant multimedia processing, such as video, and/or image signal processing or coding. However, advanced graduate studies with coursework and/or research focusing on topics of video encoding and decoding could substitute for working experience in the field.

112. I was at least a POSITA as of the earliest possible priority date of the ’877 Patent, and still am.

VIII. CLAIM CONSTRUCTION

113. As I noted above, I understand that unless claim terms are provided an express construction, the terms must be given their plain and ordinary meaning.

114. I understand that Petitioner is not advancing any constructions at this time. I have therefore applied the plain and ordinary meaning of each term in the claims throughout this declaration.

115. I also understand, however, that in separate proceedings involving the Patent Owner and a third party, the parties agreed that the term “circuit configured to” in the preamble of claims 7 and 8 should be given its “[p]lain and ordinary

meaning (*i.e., requires pre-existing programming of hardware and software to perform the cited functionality*).” Ex-1012, 2 (emphasis added).

116. Even adopting this clarification of the plain meaning would not impact my opinions because the prior art already taught a “circuit” consistent with that clarification, as I explain in further detail below.

IX. GROUNDS

A. Ground 1: Obviousness over Nakagawa in view of Sita

1. Claim 1

- a. **1[pre] – “A decoding method of a binary stream to reconstruct a current block of a current image from a reference block of a reference image reconstructed at a different size from the size of said current image, said reconstructed reference image being stored in a decoded picture buffer comprising:”**

117. In my opinion, Nakagawa discloses or suggests element 1[pre].

118. Specifically, Nakagawa discloses a “[v]ideo coding apparatus that performs predictive coding.” Nakagawa, 1:4-8. Although Nakagawa focuses on describing an *encoder*, as opposed to a *decoder*, in my opinion, a POSITA would have understood that Nakagawa describes a decoding method as claimed by describing decoding operations within the encoder. Moreover, by explaining how the decoding operations work within the encoder, a POSITA would have also

understood that Nakagawa also implicitly describes how a separate decoder compatible with Nakagawa's encoder would work.

119. To start, although they may differ in some details, encoders and decoders are typically considered the reverse of one another; the encoder performs a series of operations to convert a video signal into a stream of data, and a decoder reverses those operations to convert the stream of data back into a video signal. *See, e.g.,* Sita, ¶51 (“A decoder which recovers the image sequence from the data stream reverses the encoding process.”); Fimoff, 2:47-50 (“The decoding process implemented by the MPEG-2 decoder 30 can be thought of as the reverse of the encoding process implemented by the MPEG-2 encoder 10.”); Ex-1014, ¶4 (describing a video-encoding process, noting “[a] video decoder performs inverse operations to reconstruct the encoded video.”).

120. But predictive encoders specifically include some detail about the operation of the decoder because predictive encoders typically have a decoder loop built into them to replicate the decoding process. That is because predictive encoders use *reconstructed versions* of video images—which may differ slightly from the original versions input into the encoder due to the compression applied during encoding—to calculate motion vectors and residual errors of the next predicted frame. This reconstructed version of the video image produced by the encoder is supposed to match the reconstructed version of the video image produced by the

decoder. *See, e.g.*, Fimoff, 2:33-37 (“[t]he object of this internal loop [in the encoder] is to have the data in the reference picture memory 16 of the MPEG-2 encoder 10 match the data in reference picture memory of the MPEG-2 decoder.”); Ex-1014, ¶61 (noting how reconstructed images are used as reference frames, which may differ from the original images). If the encoder and decoder use two different images as their respective reference images (e.g., if the encoder uses the original image, but the decoder uses a reconstructed image), then the motion vectors and residual errors calculated at the encoder using one version of a reference image may not match perfectly to the different reference image at the decoder.

121. Consistent with this general understanding, Nakagawa expressly recognizes that the disclosed encoder has a decoding loop built in. Specifically, in Nakagawa’s Figure 2 (reproduced below with annotations), Nakagawa provides a diagram of the encoder with two separate outputs:

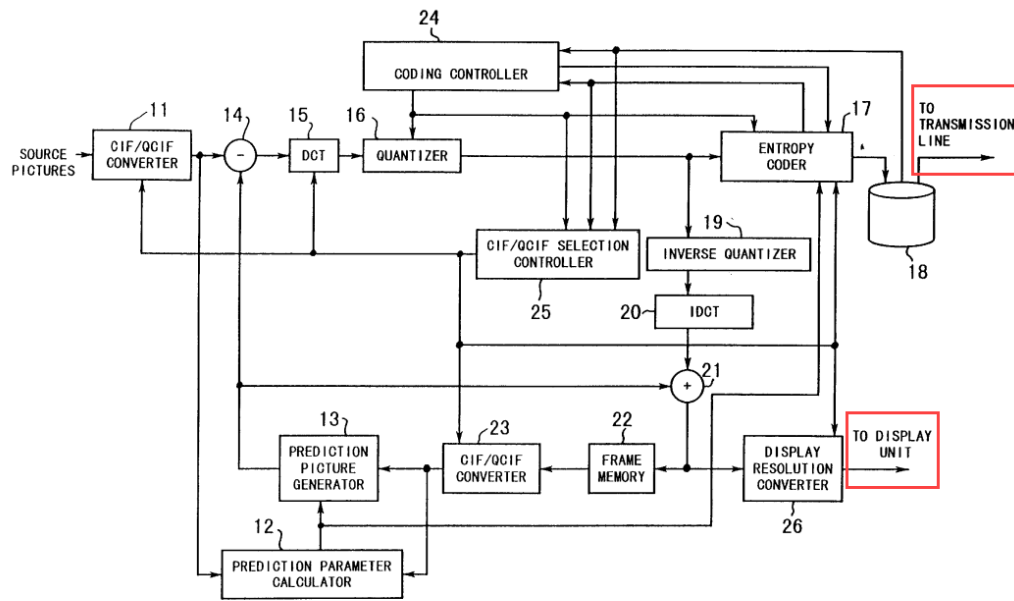


FIG. 2

The first is what would be expected as the output of an encoder: a digital bitstream for transmission (or storage in memory). The second output, however, is not typical of an encoder, but rather *a decoder*: a reconstructed video output “to [a] display unit.”

122. A POSITA would have understood that Nakagawa is showing the result of the decoding loop built into the encoder. I have identified the relevant components of the decoding loop in red, below:

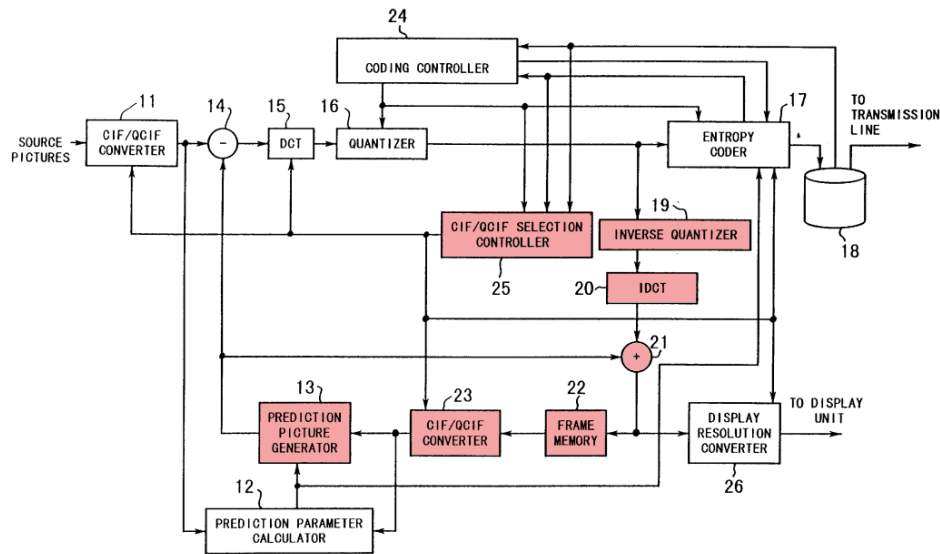


FIG. 2

Nakagawa, Fig. 2 (with annotations).

123. As Nakagawa explains, assuming a frame has already been encoded and is available as a reference in the frame memory (22), the first step to predict a subsequent frame for encoding is to calculate the motion vectors by comparing it to the reference frame using “prediction parameter calculator” (12). Nakagawa, 4:52-66. These calculated motion vectors are then applied to the reference frame to produce “a prediction picture of the current frame” (at 13), which is then subtracted from the original picture (output by 11) to calculate a “prediction error” or residual (at 14). *Id.*, 5:2-18. This residual is then encoded using the DCT transform (15) and quantizer (16). *Id.*, 5:20-41. Although both the motion vectors and (transformed and quantized) residual are sent to an entropy encoder (17) to prepare them for transmission, the residual is *also* sent into a decoding loop that immediately undoes

the quantization and transformation of the residual using an inverse quantizer (19) and inverse DCT transform (20). *Id.*, 5:42-47. This reconstructed residual is then added to the predicted picture (output by 13) to produce a “reconstructed picture [that] shows the picture of the current frame, but is not exactly the same as the original source picture because some detailed graphical information is lost in the coding process.” *Id.*, 5:47-53. This reconstructed picture is then stored in “frame memory 22” to serve as the next reference frame. *Id.*, 5:54-56. This decoding loop is how Nakagawa ensures it is calculating motion vectors and residuals that could be applied directly to the reference frames stored in the decoder, which, in my opinion, is why Nakagawa identifies the output of this decoding-loop process as being sent to a display unit. *Id.*, Fig. 2.

124. Because Nakagawa provides an express disclosure of the decoding loop within the encoder, in my opinion, a POSITA would have understood exactly how to implement a decoder (and decoding process) that complies with Nakagawa’s coding system. And even if Nakagawa’s description of the decoding loop cannot be considered an express teaching of the decoder, at a minimum, a POSITA would have understood how, and would have been motivated to configure a decoder, with a reasonable expectation of success, to operate consistent with Nakagawa’s decoding loop to ensure the bitstream output by Nakagawa’s encoder could be accurately decoded. There is little point in encoding a video into a bitstream if it cannot

subsequently be decoded. Therefore, in my opinion, a POSITA would have understood Nakagawa as at least implicitly (if not explicitly) describing a decoder too.

125. Returning to the first output of the encoder, Nakagawa also discloses that this output is a binary stream, as claimed. Specifically, Nakagawa discloses that “the coding system generates a digital video bit stream that will smoothly reproduce the original motion ... when it is decoded.” Nakagawa, 3:60-63; *see also id.*, 5:38-41 (calling the output a “bit stream”). In my opinion, a POSITA would have understood that, by disclosing an encoder that outputs, and a decoder that receives, a “bit stream,” Nakagawa is disclosing a “binary stream” as claimed because bits are just binary values (often represented as either a “0” or “1”).

126. Nakagawa also discloses that the encoder performs block-based encoding. *See* Nakagawa, 1:9-20 (“a source picture is divided into blocks of pixels”), 4:62-66 (noting to calculate the motion vectors “the source picture of the current frame [is compared] with the reconstructed picture of the previous frame on a block-by-block basis.”); 5:13-26 (noting residuals are encoded a block at a time); *see also id.*, 4:43-51.

127. In my opinion, based on Nakagawa’s description of the encoder, a POSITA would have understood that the decoder would reconstruct a block of a current frame from a reference block of a reference frame, as claimed. Specifically,

because the motion vectors and residuals are being encoded on a block-level basis, a POSITA would have appreciated that when the motion vectors and residuals calculated at the encoder are decoded from the bit stream at a decoder, the motion vectors and residual would be used to reconstruct the current frame from a reference frame on a block-level basis as well. *See, e.g.*, Nakagawa, 5:2-12, 5:42-59 (describing reconstruction process in the encoder's decoder loop). This is just reversing the operation of the encoder.

128. Nakagawa also discloses reconstructing a current frame using a different-sized reference frame. Specifically, Nakagawa discloses that each frame can be encoded to have at least one of two sizes: CIF (which stands for "common intermediate format" and is an image with 352x288 pixels) or QCIF (which stands for "quarter CIF" and is an image with 176x144 pixels). Nakagawa, 4:43-51. Nakagawa explains that the encoder can switch between these two sizes in a video stream based on factors that are evaluated after each frame is encoded. *See, e.g., id.*, 4:2-6, 7:52-54. Nakagawa also describes how, in some instances, a block from a current image (e.g., CIF) will be reconstructed using a block from a reference image stored at a different size (e.g., QCIF) than the current image, and explains how to resample the reference image to match the resolution of the current image. *See, e.g., id.*, 6:15-40.

129. In my opinion, a POSITA would have understood that if the current and reference images have different sizes at the encoder, they would also have different sizes when being reconstructed at the decoder. That is because Nakagawa concerns switching between different image resolutions during encoding to avoid over-compressing images or overloading the buffer, yet still being able to reconstruct the current image even when the reference image is a different size. *See, e.g.*, Nakagawa, 3:22-34, 6:3-40. That means the encoder can send both CIF and QCIF images to the decoder for reconstruction during a single video stream. Accordingly, just like with the encoder, sometimes the reconstructed image will be CIF, and other times it will be QCIF, and the decoder must also have a way to resample a reference image so that it always matches the size of the current image. *See, e.g., id.*, 4:66-5:1 (“Note that the pictures of the two consecutive frames are configured to have the same resolution for easy comparison.”).

130. Finally, Nakagawa discloses storing reconstructed reference images in a picture buffer, which Nakagawa calls a “buffer” or “frame memory.” *See, e.g.*, Nakagawa, 4:58-61 (“A reconstructed (or decoded) picture of the previous frame stored in a frame memory 22”); *see also id.*, 2:55-62, 5:54-56. A POSITA would have understood that Nakagawa’s decoder would also have a frame memory to store reference images for reconstructing current images as well. This is evident from the fact that Nakagawa’s “frame memory” is part of the decoder loop. *See id.*, Fig. 2.

131. A POSITA would have also understood that including a frame memory or a buffer in the decoder is conventional practice to store reference frames for reconstructing current images. *See, e.g.*, Sita, Fig. 2B (showing a decoder with reference frame memory 222 in the same relative position as frame memory in Nakagawa's decoder loop); Fimoff, Fig. 5 (same, showing "[r]educed resolution reference pictures" memory 106); Ex-1014, Figs. 2, 4 (showing "reference frame store" in both the encoder (Fig. 2) and decoder (Fig. 4)).

- b. 1[a] – "motion compensating said reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,"**

132. In my opinion, the combination of Nakagawa and Sita teaches element 1[a].

133. Nakagawa discloses a process to calculate motion vectors at the encoder by "[c]omparing the source picture of the current frame with the reconstructed picture of the previous frame on a block-by-block basis." Nakagawa, 4:62-65. Nakagawa also discloses that, as part of the decoder loop, a "prediction picture generator" receives the motion vectors and "produces a prediction picture of the current frame" by applying the calculated motion vectors to the reference picture, *id.*, 5:4-12, which would be understood by a POSITA as being a motion-compensation process.

134. Because it is possible for the reference image to have a different size than the current image being predicted as part of the motion-compensation process in the decoder loop, Nakagawa discloses a resolution converter (“CIF/QCIF converter”) that resamples the reference image so that its size will match that of the current image. *See id.*, 6:15-40, *id.*, 5:4-12 (describing application of motion vectors), Fig. 2 below (showing a CIF/QCIF converter (23) for resampling, and prediction picture generator (13) to apply the motion vectors calculated by prediction parameter calculator (12)):

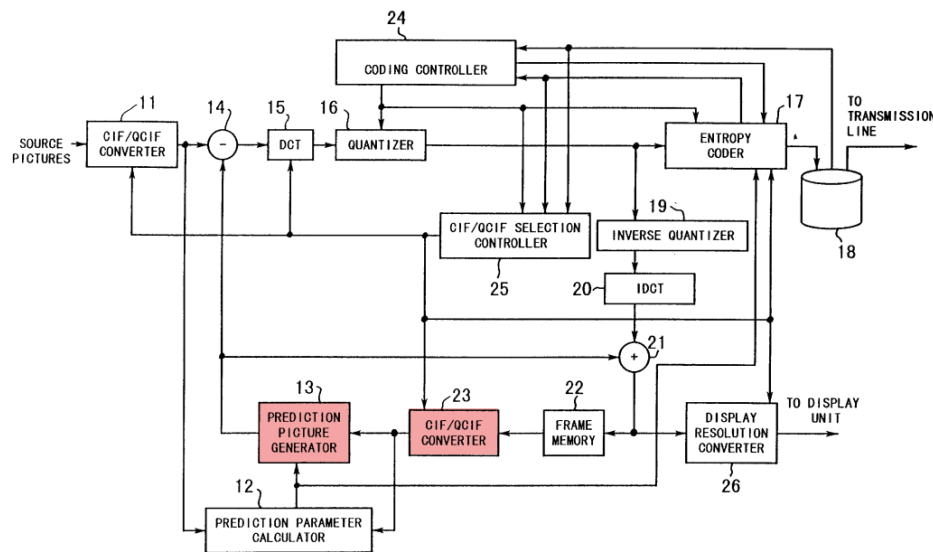


FIG. 2

135. Although Nakagawa does not disclose how exactly the CIF/QCIF converter operates with respect to the motion-compensation process, a POSITA would have recognized that other art had faced similar problems in the past and already provided that explanation. Specifically, in my opinion, a POSITA would

have been well-aware that a similar situation arose when television networks began broadcasting high-definition television signals, yet standard-definition televisions were still in widespread use. To ensure those legacy televisions could display high-definition television content, those in the art had figured out how to downsample high-definition video signals to standard definition without overloading legacy buffers in those televisions.

136. Sita, for instance, is one of several who provided a solution to this exact problem. Sita teaches a video decoder that operated in accordance with predictive video-coding standards, like Nakagawa. *See* Sita, ¶2 (citing MPEG-2); Nakagawa, 1:10-13 (citing H.261, MPEG-1, and MPEG-2). Unlike Nakagawa, Sita focuses on the decoder, and teaches that when a current (high-definition) frame needs to be reconstructed from a (standard-definition) reference frame, a joint resampling (226) and motion-compensation-interpolation filter operation (228) (collectively, filter 224) is employed in which a reference block is motion compensated by applying the filter. Sita, ¶¶82, 88-89.

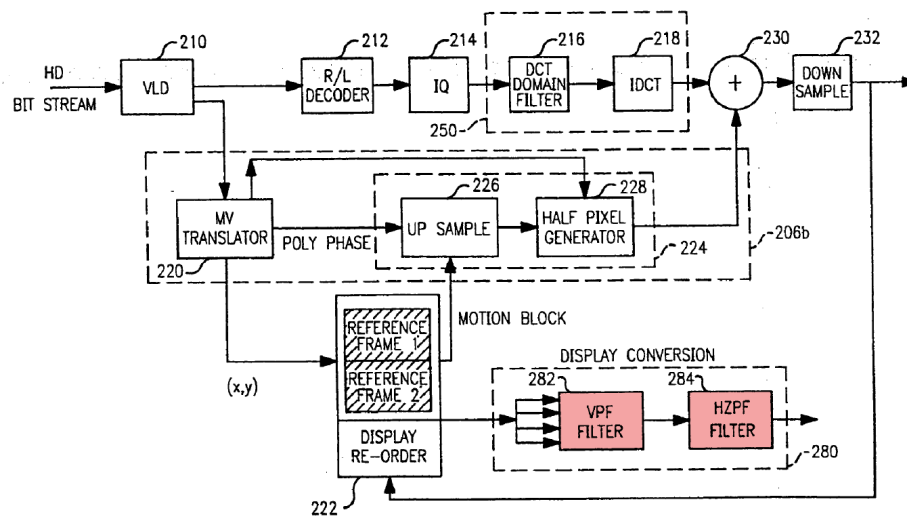
137. Although Sita represents filter (224) as a single filter and does not expressly show separate horizontal and vertical filter operations, as claimed, *see* Sita, Fig. 2B, in my opinion, a POSITA would have understood that filter (224) would apply separate horizontal and vertical filter operations, or at the very least would have known how and been motivated to configure Sita that way.

138. Specifically, Sita explains how MPEG-2 can provide video in several different image sizes with different numbers of pixels in both the horizontal and vertical direction. *See, e.g.*, Sita, ¶66 (Table 1 showing nine different size formats). Sita also explains that these various image sizes can all be reformatted to have different, smaller sizes. *Id.*, ¶67 (Table 2 showing downconversion of all formats to 525P/I format). Because the downconversion would change the number of pixels in both the horizontal and vertical directions, in my opinion a POSITA would have understood Sita to teach at least by Tables 1 and 2 that resampling must be performed in both the horizontal and vertical directions to reformat the images.

139. Moreover, when Sita describes the filtering operation, Sita primarily focuses on scaling in only the horizontal direction as an example. *See, e.g., id.*, ¶¶92, 93, Figs. 2C, 2D. But even Sita acknowledges that “[o]ne skilled in the art could *easily* extend the following discussion to motion translation in the vertical direction if desired.” *Id.*, ¶95 (emphasis added). I agree. And, in my opinion, a POSITA would have had a desire to perform filtering operations in both the horizontal and vertical directions—and not just the horizontal direction—to minimize the size of reference images that need to be stored in memory compared to images that are resampled in only one direction. Performing filtering in both directions also avoids distorting images and/or inducing artifacts in only one direction. *See* Ex-1017, 13:16-33 (explaining how a 3x3 pixel array “is first upsampled in the horizontal direction to

become a 5x3 pixel array” and then “the 5x3 pixel array...is upsampled in the vertical direction to become the final 5x5 pixel array”).

140. Finally, Sita also shows that as part of a *separate* filtering process, reconstructed images may be sent through a final resampling filter to adjust them to a resolution that matches the display. *Id.*, Fig. 2B, ¶¶81. An annotated version of Sita’s Figure 2B is provided below, which identifies those filters:



This explicitly shows separate resampling filter operations being applied in a vertical direction followed by a horizontal direction.

141. Based on all of these teachings, in my opinion, a POSITA would have understood that Sita's filter (224) would likewise perform its filtering operation using separate horizontal and vertical filters. In fact, it was well known at the time that two-dimensional filtering operations of this type could be implemented as either a single two-dimensional filter or, equivalently, as two iterated one-dimensional

filters, but that using two iterated one-dimensional filters would consume less computational resources. *See, e.g.*, Ex-1015, 6:14-26 (“[A]pplying ... a vertical one-dimensional filter and afterwards a second horizontal one-dimensional filter to the same region ... provides a two-dimensional filter effect, however, with a reduced complexity compared to an application of a two-dimensional filter.”). Thus, even if Sita cannot be read as teaching a filter (224) that applies separate horizontal and vertical filter operations, a POSITA would have been motivated and reasonably expected to implement it as such to conserve computational resources.

142. Moreover, even though Sita shows as part of the separate display resampling that a vertical filter is applied before a horizontal filter, in my opinion, a POSITA would have appreciated that the order of the filters is of little, if any, significance in such two-dimensional resampling operations. If a POSITA were to employ two one-dimensional filters instead of a single two-dimensional filter, a POSITA would have understood that only two options were available for the ordering of the filters: (1) horizontal filtering followed by vertical filtering or (2) vertical filtering followed by horizontal filtering. The selection of a specific sequencing of filtering operations—all of which would have produced similar results—would have been well within the implementation expertise of a POSITA. Even the ’877 Patent recognizes that. *See* ’877 Patent, 4:33-36 (“It is immaterial to apply first the horizontal filter on the lines then the vertical filter on the columns or

conversely to apply first the vertical filter on the columns then the horizontal filter on the lines.”).

143. In my opinion, therefore, a POSITA would be motivated to incorporate Sita’s filtering operations into Nakagawa’s decoding loop (and also, by extension, a compatible decoder) to convert, e.g., QCIF images to CIF images (and vice versa) as part of the motion-compensation process. That includes performing separate horizontal and vertical filtering operations not only to conserve computational resources, *see, e.g.*, Ex-1015, 6:14-26, but also because Sita solved the exact problem of performing upsampling during motion compensation to match resolution (size) of a reference block to the current block. Sita fills in the gap left by Nakagawa’s silence regarding a particular implementation of the filter operations.

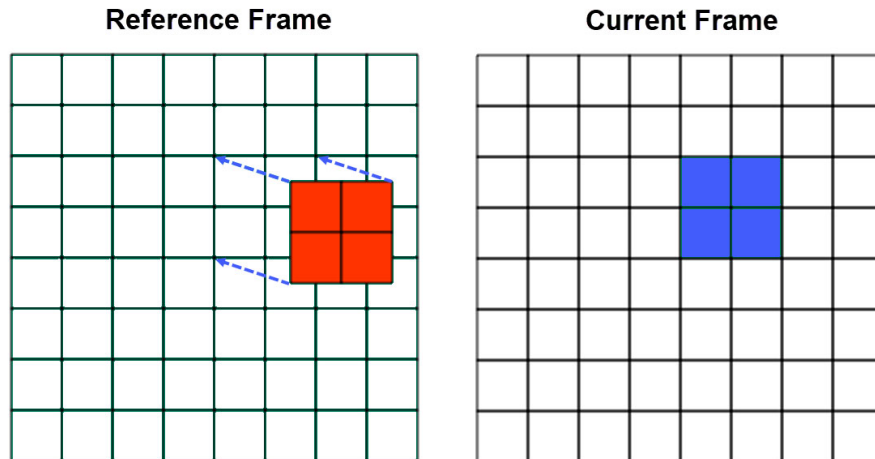
144. In my opinion, a POSITA would have been further motivated to combine Nakagawa and Sita because Sita would have made it easier to implement other coding schemes as part of Nakagawa’s invention. Throughout its disclosure, Nakagawa explains the invention, and particularly the embodiment of Figure 2, using “ITU-T standard H.261” as exemplary, and adopting its video size formats (CIF and QCIF) to explain the invention. Nakagawa, 4:43-51. In my opinion, a POSITA would have understood that Nakagawa was not limiting the disclosure to H.261, however, because none of the video coding blocks in the embodiment of Figure 2 are limited to an H.261 coding scheme. *See* Nakagawa, 1:10-13

(recognizing that “H.261 and ... MPEG-1 and MPEG-2, for example, are well-acknowledged international standards for motion picture coding techniques, which employ hybrid coding algorithms.”), 10:3-4 (“The foregoing [disclosure] is considered as illustrative only of the principles of the present invention.”).

145. A POSITA would have also understood that newer video coding standards such as MPEG-1 or MPEG-2 provided their own benefits, such as better-quality video, more options for video resolutions (instead of just CIF/QCIF for H.261), the availability of numerous implementation choices due to their wider adoption, and, critically, more-accurate motion compensation by using half-pixel (half-pel) motion-compensation-interpolation procedures rather than the full-pixel (full-pel or integer-pel) motion compensation used by H.261. *See, e.g.*, Ex-1029, 68 (explaining similarities between H.261 and MPEG-1, but noting how “[c]omparing H.261 and MPEG1, [motion-compensated interpolation in MPEG1] this is a major new function added to the video codec to achieve further (better) compression.”).

146. Specifically, it may occur that, between two temporally adjacent video images, an object in a video moves by a distance that is not exactly an integer number of whole pixels. For example, an object such as a car or an airplane might move by 2.5 pixels between two adjacent video images. An example of an object that moves a fractional number of pixels is shown below, where the current frame (shown at

right) searches for an object (the blue pixels) that spans an area *between* pixels in the reference frame (on the left, in red).



To accurately compensate for the motion of the object between the reference frame and the current frame, the motion vectors would capture a movement of $1 \frac{1}{2}$ pixels to the left, and $\frac{1}{2}$ pixels up, as illustrated by the blue arrows. Motion compensation would then apply interpolation filters with half-pixel resolution to calculate the values of the red pixels—because they lie *between* the actual pixels of the reference frame. Allowing motion vectors with half-pixel resolution can therefore lead to more accurate predictions because capturing the motion of an object is not limited to full-pixel displacements.

147. In my opinion, therefore, a POSITA would have been motivated to configure Nakagawa's system to implement these other coding schemes, which Nakagawa itself expressly identifies as a possible variation to its disclosed QCIF/CIF resolution examples. *See* Nakagawa, 1:10-13 (expressly identifying

H.261, MPEG-1, and MPEG-2 as “well-acknowledged international standards for motion picture coding techniques.”), 4:43-44 (using H.261 to describe the first embodiment).

148. As opposed to H.261, which is limited to integer-pixel values for motion vectors, MPEG-1 and MPEG-2 adopted half-pixel resolution motion-vector values because this provided improved image quality for a given level of compression and improved compression for a given level of image quality. And upon configuring Nakagawa to implement MPEG-1 or MPEG-2, a POSITA would have further appreciated that Sita’s filter was already compliant with those coding standards, including their implementation of half-pixel motion compensation operations. *See* Sita, ¶2 (describing an MPEG-2 compliant scheme). Thus, Sita’s filtering operations would have made it *easier* to implement other coding schemes in Nakagawa’s system.

149. In my opinion, a POSITA would have also had a reasonable expectation of success in incorporating Sita’s filtering operations into Nakagawa’s decoding loop (and also, by extension, a compatible decoder). Both Nakagawa and Sita perform similar operations: resampling and motion-compensation interpolation. The only significant difference is that Nakagawa performs these as two separate steps whereas, as I will discuss in detail subsequently, Sita performs them together as a single joint filter for both resampling and motion compensation interpolation. The

modification would have involved little more than computer coding to operate on simple data structures, such as arrays, which would have been well within the skill of a POSITA.

c. 1[b] – “decoding, for the current block, a residue block, and”

150. In my opinion, Nakagawa discloses or suggests element 1[b].

151. Specifically, Nakagawa expressly discloses that the encoder generates a residue block, which Nakagawa calls a “prediction error signal.” Nakagawa, 5:13-17 (“The prediction error calculator 14 produces a prediction error signal by calculating differences between the current frame picture ... and the prediction picture.”).

152. In my opinion, a POSITA would have understood that Nakagawa’s “prediction error signal” is a “residue block,” as claimed, because the “prediction error signal” captures the differences between the current frame picture and the prediction picture. Nakagawa, 5:13-17. That is consistent with how “residuals” are described throughout the prior art. *See, e.g.,* Sita, ¶¶77-78 (explaining IDCT processor outputs “differential luminance or differential chrominance components of the predicted macroblock,” which “correspond[s] to a *residual* component of the present predictively encoded frame.”) (emphasis added); Fimoff, 1:57-64 (“The output of the subtractor 12 *is an error, or residual*, that represents the difference

between the predicted macroblock and the new macroblock being encoded.”) (emphasis added). This is also consistent with how the ’877 Patent describes calculating the “residue block.” *See* ’877 Patent, 9:5-11 (“the residue block is obtained by subtracting the prediction block pixel from the current block”).

153. Furthermore, Nakagawa discloses that, as part of the decoder loop, a bitstream of “quantized transform coefficients” are decoded “to reproduce the prediction error signal.” Nakagawa, 5:42-47. Because this is described as part of the decoder loop, in my opinion, a POSITA would have understood that a decoder implemented according to Nakagawa would have likewise decoded a “prediction error signal” because decoding the residual is part of the conventional process for reconstructing a current block based on a predicted reference block. *See, e.g.*, Nakagawa, 5:47-53 (“A reconstructed picture generator 21 reconstructs a picture by adding the prediction picture produced by the prediction picture generator 13 and prediction error signal from the IDCT processor.”); Sita, ¶78 (describing a decoder in which “[t]he Motion Compensation Processor 206 receives a previously decoded macroblock from memory 199 responsive to the motion vector, and then adds the previous macroblock to the current IDCT macroblock (corresponding to a residual component[]) ... to produce the corresponding macroblock of pixels for the current video image.”); Fimoff, 6:61-62 (describing a process of “adding prediction reference pixels to the intermediate residual values, as appropriate, to form

reconstructed pixels.”). In other words, the residual is calculated at the encoder by subtracting the predicted image from the current image, and at the decoder, the current image is reconstructed by adding the residual to the predicted image.

154. Moreover, a POSITA would have also appreciated that the ’877 Patent was not reciting any new teachings as to the use of a residual, but was just following standard practice of using a residual to reconstruct the current block, as had long been done before.

d. 1[c] – “reconstructing the current block from said residue block and from said motion compensated reference block,”

155. In my opinion, Nakagawa discloses or suggests element 1[c].

156. As I previously explained in Sections IX.A.1.a (element 1[pre]) and IX.A.1.b (element 1[a]), Nakagawa discloses that, within the decoding loop, a “prediction picture generator” applies the motion vectors to a reference picture to produce a predicted picture on a block-by-block basis. And as I also explained in explained in Section IX.A.1.c (element 1[b]), Nakagawa also discloses that, within the decoding loop, “[a] reconstructed picture generator 21 reconstructs a picture by adding the prediction picture produced by the prediction picture generator 13 and the prediction error signal from the IDCT processor 20.” Nakagawa, 5:47-53.

157. For the same reasons I discussed above, *see* Sections IX.A.1.a (element 1[pre]), IX.A.1.c (element 1[b]), by describing these operations in the

context of a decoding loop, a POSITA would have understood that these operations would also be performed in a decoder implemented according to Nakagawa.

158. Moreover, reconstructing a current block by combining a residue block and motion-compensated reference block (the prediction) is the conventional process to reconstruct a decoded image at a decoder, *see* Section IX.A.1.c (element 1[b]), and is consistent with the operations performed by Sita's decoder. Sita, ¶78 ("The Motion Compensation Processor 206 receives a previously decoded macroblock from memory 199 responsive to the motion vector, and then adds the previous macroblock to the current IDCT macroblock (corresponding to a residual component of the present predictively encoded frame) ... to produce the corresponding macroblock of pixels for the current video image[.]").

159. For these reasons, in my opinion a POSITA would have understood that Nakagawa discloses reconstructing the current block by combining a residue block (Nakagawa's "prediction error signal") with a motion-compensated reference block (the "prediction picture"), as claimed.

- e. **1[d] – “wherein said single vertical filter GF_v applied on a pixel s is such that $GF_v(s)=MCIF_v(SCF_v(s))$, where MCIF_v is a vertical motion compensation interpolation filter and SCF_v is a vertical resampling filter, MCIF_v and SCF_v being applied jointly and”**

160. In my opinion, the combination of Nakagawa and Sita teaches element 1[d].

161. To first understand the scope of the claim, although this claim element uses the equation $GF_v(s)=MCIF_v(SCF_v(s))$ and abbreviations “MCIF_v” and “SCF_v,” in my opinion a POSITA would have understood that this notation and these abbreviations refer to the definition of the “single filter” (GF_v) as a mathematical composition of (i.e., as the joint application of) the individual filters SCF_v and MCIF_v. Specifically, a POSITA would have understood that the single vertical filter (GF_v) comprises the joint application of a vertical motion compensation interpolation filter (MCIF_v) and a vertical resampling filter (SCF_v) combined into one joint filtering operation. *See* ’877 Patent, 5:57-67 (“According to the invention, the horizontal filters MCIF_H and SCF_H are grouped into a single filter GF_H applied on the lines of a pixel block” using, e.g., equation $GF_H = MCIF_H \circ SCF_H$ (with the same notation used for vertical filter “V”), “where $(f \circ h)(x) = f(h(x))$.”). As Interdigital’s expert explained during the evidentiary hearing in ITC Case No. 337-TA-1373 initiated against Lenovo (United States) Inc., et al., the joint filter

essentially uses an equation that applies weights to pixel values in a reference frame to calculate new pixel values that represent both a resampled and motion-compensation interpolated reference frame, which would be the same outputs as if separate resampling and motion-compensation interpolation equations had been used. *See, e.g.*, Ex-1067, 525:11-527:21. I agree with this view of the claimed equation.

162. With this understanding, as explained in Section IX.A.1.b (element 1[a]), Nakagawa and Sita together teach “applying...a single vertical filter GFv successively ... on the columns of pixels of said reference block” as claimed.

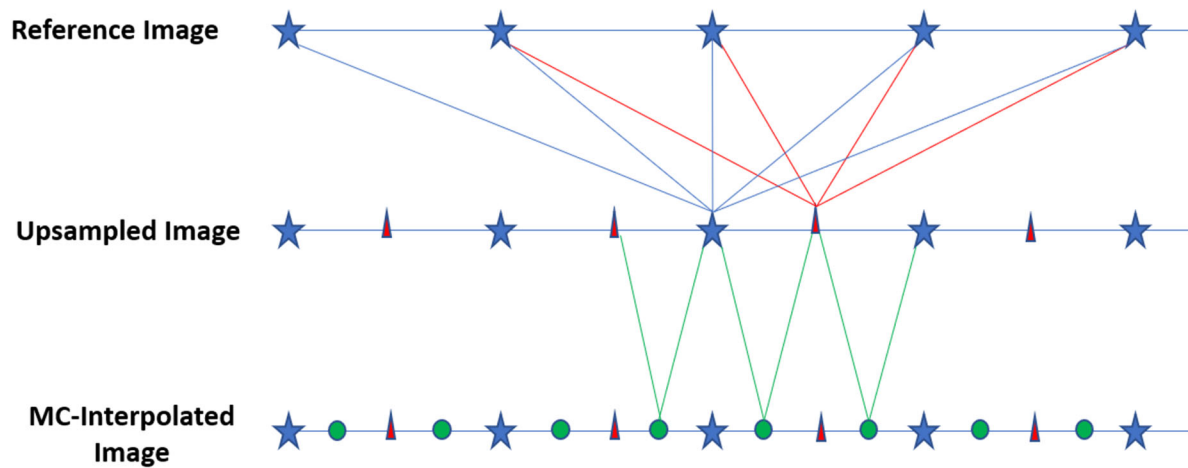
163. Nakagawa also discloses that, as part of the decoding loop, a vertical resampling operation is performed. *See, e.g.*, Nakagawa, 4:46-51 (noting how CIF pictures have a vertical resolution of 288 pixels, and QCIF pictures have a vertical resolution of 144 pixels), 6:15-40 (describing a resampling operation to convert CIF to QCIF, and vice-versa). Nakagawa further discloses that, after resampling, a motion-compensation operation is performed. *Id.*, Fig. 2 (showing CIF/QCIF converter 23 followed by prediction picture generator 13), 5:2-12.

164. However, in my opinion, a POSITA would have appreciated that Nakagawa does not explain exactly how the CIF/QCIF convertor and prediction picture generator operate, and does not expressly disclose performing motion-compensation interpolation (because Nakagawa implements H.261, which

implements only full-pixel motion vectors). Accordingly, a POSITA would have looked to other references to provide additional guidance. To that end, a POSITA would have been aware of Sita, which provides a description of a decoder that, just like Nakagawa, implements reference-frame upsampling as part of a motion-compensation operation, and performs motion-compensation interpolation to account for MPEG-2's half-pixel motion vectors. *See* Section IX.A.1.b (element 1[a]). Specifically, Sita describes a specific, e.g., 3:1 or 2:1 upsampling of a reference image to match the size of the current image as part of the motion-compensation operation. *See* Sita, ¶¶108, 109.

165. Although Sita depicts the motion-compensation filter (224) as two steps—"upsampling" (226) followed by motion-compensation-interpolation (228)—and sometimes refers to it that way, *see, e.g.*, Sita, ¶¶89, 99, a POSITA would have understood these descriptions are for ease-of-reference, and not because Sita actually implements these two operations separately.

166. Provided below is an example illustrating Sita's upsampling and interpolation operations (224) using a 2:1 upsampling ratio as an example, where the individual upsampling and motion-compensation interpolation operations are shown separated out:



In this figure, the top row shows the original pixels of the reference image that are input to the upsampling and motion-compensation interpolation operations. The middle row shows the intermediate pixels of the upsampled image that result from application of the 2:1 upsampling operation. The bottom row shows final output pixels of the upsampled and motion-compensation interpolated image that result from application of both 2:1 upsampling and half-pixel motion-compensation interpolation. In all three rows, the blue stars represent pixel location corresponding to the pixel locations in the original reference image. The red triangles represent new pixel locations that result from the 2:1 upsampling operation and therefore are positioned halfway between each of the blue stars (original pixel sites from the input reference image). The green circles represent new pixel locations that result from the half-pixel resolution motion-compensation interpolation operation; they fall at half-pixel locations in the 2:1 upsampled image depicted in the middle row (which correspond to quarter-pixel locations with respect to the original reference picture

pixel locations shown in the top row for this example with 2:1 upsampling ratio) and are therefore positioned halfway between each of the reference and upsampled subpixel locations. This process uses a five-tap upsampling filter operation, meaning five pixels from the reference image are weighed to create each pixel in the upsampled image, and a two-tap interpolation filter, meaning two pixels from the upsampled image are weighed to calculate each final upsampled and motion-compensation interpolated pixel value.

167. At the first stage of the example above, a reference image is upsampled with the five-tap filter using a 2:1 upsampling ratio. This doubles the number of pixels. At the second stage, half-pixel-resolution motion-compensation interpolation is performed, which again doubles the number of pixels relative to the upsampled image.

168. But Sita does not limit implementation of the Motion Block Generator (224) to sequentially applying the resampling and interpolation filters as I have just illustrated in the preceding example. Instead, Sita provides a detailed description of the operations, explaining that, in practice, the upsampling and motion-compensation interpolation operations can be applied jointly. *See* Sita, ¶¶110-111. That is, Tables 7B and 7C provide specific filter coefficients (for 3:1 upsampling or 2:1 upsampling, respectively) for a single joint filter to calculate both the upsampled and half-pixel interpolation values in a single combined filtering operation based

only on the pixels in the original reference image and without calculating or storing the intermediate values I showed in the middle row of the figure above. An annotated version of Table 7B is provided below, with the filter coefficients for each subpixel position generated by the filter identified separately:

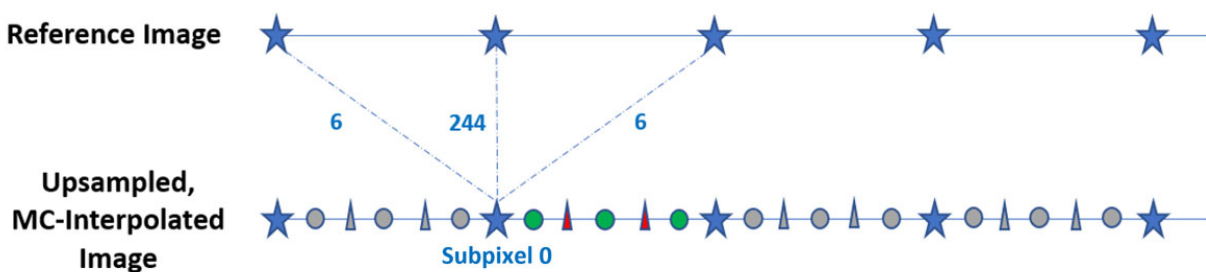
TABLE 7B

	Sub-pixel 0	Sub-pixel 1	Sub-pixel 2	Sub-pixel 3	Sub-pixel 4	Sub-pixel 5
Filter Coeff.	6 244 6 0 0	-18 223 51 0 0	-42 202 96 0 0	-21 149 149 -21 0	96 202 -42 0 0	51 223 -18 0 0
Reference	$x^* - 1$	$x^* - 1$	$x^* - 1$	$x^* - 1$	x^*	x^*
Phase	01	00	10	01	00	10
Half Pixel	0	1	0	1	0	1

169. In Table 7B, Sita discloses the filter coefficients of a combined filter for performing a 3:1 upsampling operation jointly with a 2:1 motion-compensation-interpolation operation, resulting in six possible subpixel positions relative to each reference pixel in the reference frame. Referring to annotated Table 7B above, because the final subpixel values are calculated at six different fractional-pixel locations with respect to the original reference-pixel locations, six different sets of filter coefficients are listed in six columns in the “Filter Coeff.” row. In the next row labeled “Reference” the notation “ $x^* - n$ ” indicates the number “ n ” of pixels needed to the left of (the earliest) upsampled pixel location (the red triangles in the previous

picture). *See* Sita, ¶107 (explaining “ $x*-n$ ” represents “a number of left pixels which are needed in addition to the pixels in the upsampled block for the upsampling process.”), Table 7A. The next two rows “Phase” and “Half Pixel” together identify the relative fractional pixel location using a bitwise notation. *See* Sita, ¶113. This type of filter that uses a different set of filter coefficients for computing the output pixel value for each output subpixel location is called a *polyphase filter* and it would have been well-known to a POSITA at the time.

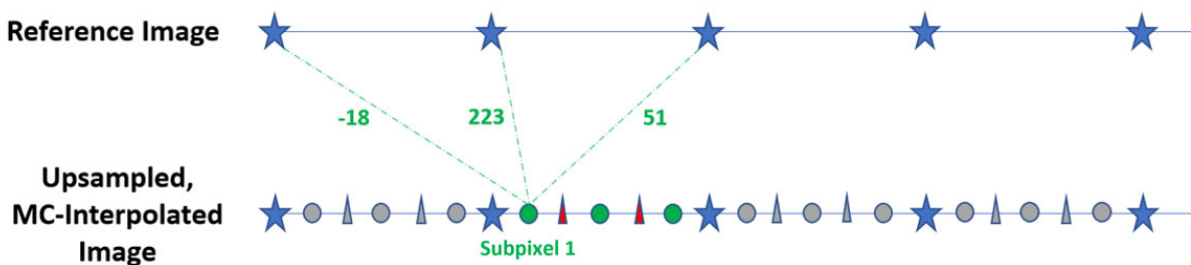
170. These filter coefficients are applied to the original pixel values of the reference image in the joint five-tap filter as shown in Table 7B to perform both upsampling and motion-compensation interpolation as a single combined filtering operation. A representation of how each upsampled and motion-compensation interpolated output pixel value is calculated at each subpixel location is provided below, beginning with Subpixel 0.



Each filter coefficient (the blue numbers in the figure above) is applied to its respective reference-pixel value input into the five-tap filter (the blue stars in the upper row of the figure above) to generate the new upsampled and motion-

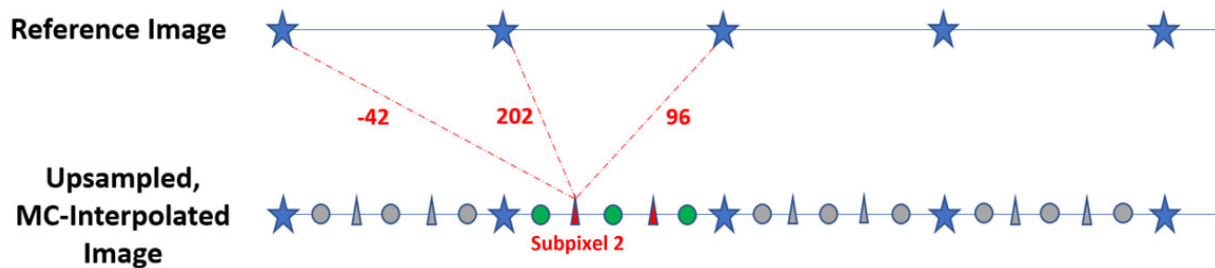
compensation-interpolated output pixel value for subpixel location 0 (shown as a blue star “Subpixel 0” in the lower row of the figure above). The (second) reference pixel closest to the new Subpixel 0 location is given the most weight (244), with much smaller contributions (6) from neighboring reference pixels. And even though the final two reference-pixel values to the far right are input into the filter, they are not weighed for this particular output pixel because their filter coefficients are zero.

171. A similar process using the second column of filter coefficients (boxed in green in annotated Table 7B above) is then applied to calculate the upsampled and motion-compensation-interpolated output pixel value for subpixel location 1 (shown as a green circle “Subpixel 1” in the lower row of the figure below), which is positioned at half-pixel resolution with respect to the pixel sites implied by the 3:1 upsampling ratio:

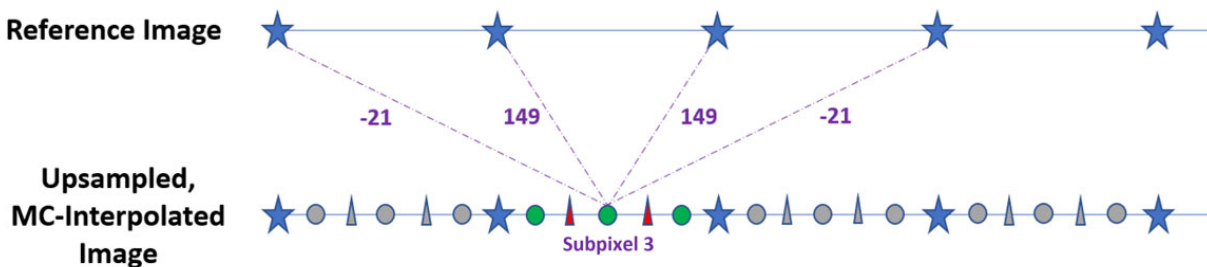


A similar process is then applied using the third column of filter coefficients (boxed in red in annotated Table 7B above) to calculate the upsampled and motion-compensation-interpolated output pixel value for subpixel location 2 (shown as a red

triangle “Subpixel 2” in the lower row of the figure below), which is positioned at the second full-pixel resolution pixel site implied by the 3:1 upsampling ratio:

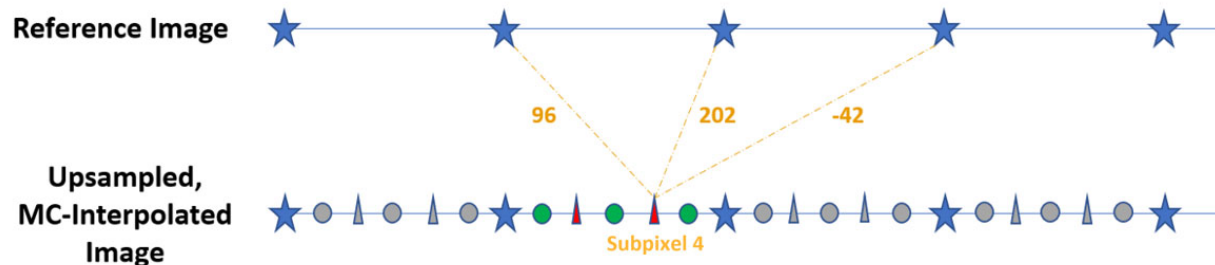


172. Next, the filter coefficients from the fourth column (boxed in purple in annotated Table 7B above) are used to calculate the upsampled and motion-compensation-interpolated output pixel value for subpixel location 3 (shown as a green circle and labeled “Subpixel 3” in the lower row of the figure below), which is positioned at a half-pixel resolution pixel site in the upsampled and motion-compensation-interpolated output image. In calculating the filter output value for Subpixel 3, an additional reference pixel (the fourth reference pixel to the right) is given a nonzero weight in addition to the previous three reference pixels as depicted in the figure below:

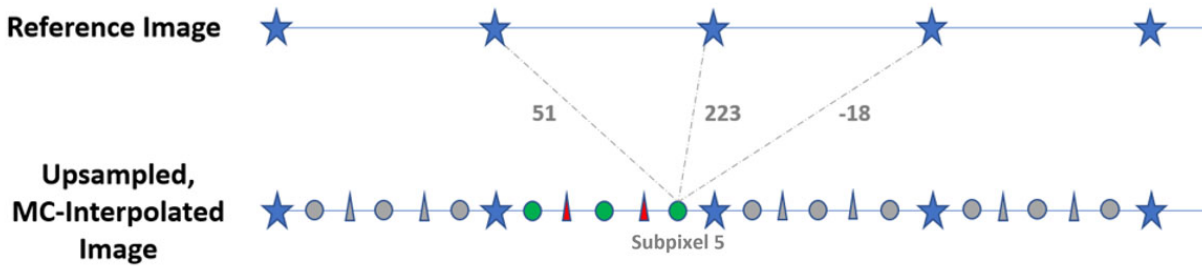


Because the site of Subpixel 3 sits exactly halfway between the two sets of input reference pixels on either side, the reference pixels on each side are weighted identically (either 149, or -21, respectively), reflecting their relatively equal contributions to the upsampled and interpolated subpixel output value.

173. For the upsampled and motion-compensation-interpolated output pixel value at subpixel location 4 (shown as a red triangle “Subpixel 4” in the lower row of the figure below), Sita’s combined filter uses the fifth column of filter coefficients (boxed in red in annotated Table 7B above):



For Subpixel 4, which is positioned at the third full-pixel resolution pixel site implied by the 3:1 upsampling ratio in this example, the first input reference pixel value (to the left) no longer receives a nonzero weight and nonzero filter coefficients are applied only to the next three input pixel values from the reference image. This is confirmed in Table 7B by the identification of the “Reference” pixel, which switches from “ $x*-1$ ” to “ $x*$ ” in the first table row appearing below the filter coefficients. Like Subpixel 4, Subpixel 5 is then determined without contribution from the first reference pixel (because it too identifies the left-most pixel used as “ $x*$ ”):



174. In sum, Sita expressly teaches a joint filter that operates directly on the input reference pixels (blue stars) and omits any intermediate calculation or storage of the upsampled pixel values before outputting the final half-pixel-resolution upsampled and motion-compensation interpolated output pixel values. In other words, the filter coefficients Sita discloses combine the filtering performed by the upsampling operation and the motion-compensation interpolation operation in a single joint filtering operation.

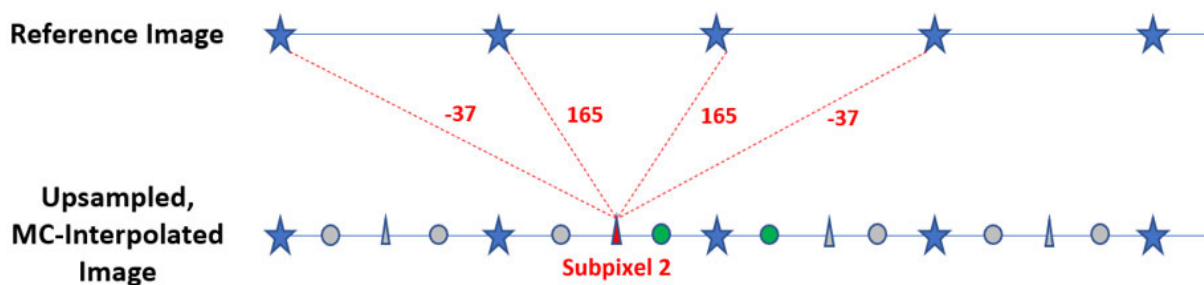
175. A similar pattern applies to Sita's combined 2:1 upsampling and half-pixel motion-compensation interpolating filter disclosed in Table 7C. An annotated version of Table 7C is provided below, with the filter coefficients for each subpixel output position identified by a separately colored annotation:

TABLE 7C

	Subpixel 0	Subpixel 1	Subpixel 2	Subpixel 3
Filter Coeff.	3 7 236 7 3	2 -15 200 86 -17	-37 165 165 -37 0	-17 86 200 -15 2
Reference	$x^* - 2$	$x^* - 2$	$x^* - 1$	$x^* - 1$
Phase	00	00	01	01
Half Pixel	0	1	0	1

Here, unlike Table 7B that applied nonzero filter coefficients to only some of the input reference pixels at each subpixel position, every one of the reference pixel values that is input into the five-tap filter is used with a nonzero weight to calculate the subpixel values at each position (except Subpixel 2, which is symmetrically at a half-way pixel location as shown below).

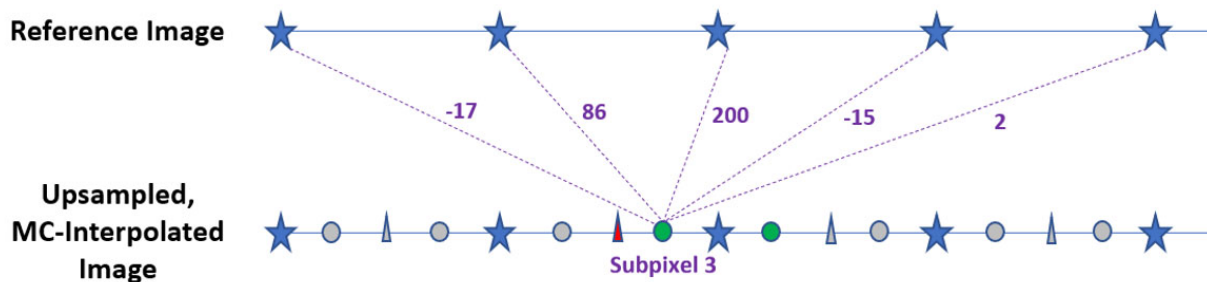
176. I begin by illustrating the calculation of Subpixel 2 as shown in the figure below:



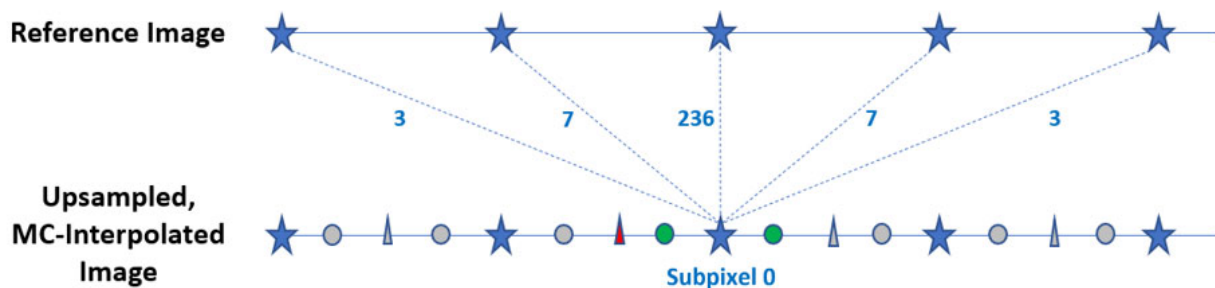
I begin with Subpixel 2 because, unlike Subpixels 0 and 1 which require two *additional* pixels to the left of the center reference pixel, as reflected by the notation “ x^*-2 ” in the row labeled “Reference” in Table 7C, the filter phase for Subpixel 2

requires only one additional pixel to the left (as reflected by the notation “ $x*-1$ ” in Table 7C) which appears as the reference pixel furthest to the left in the upper row of the figure above.

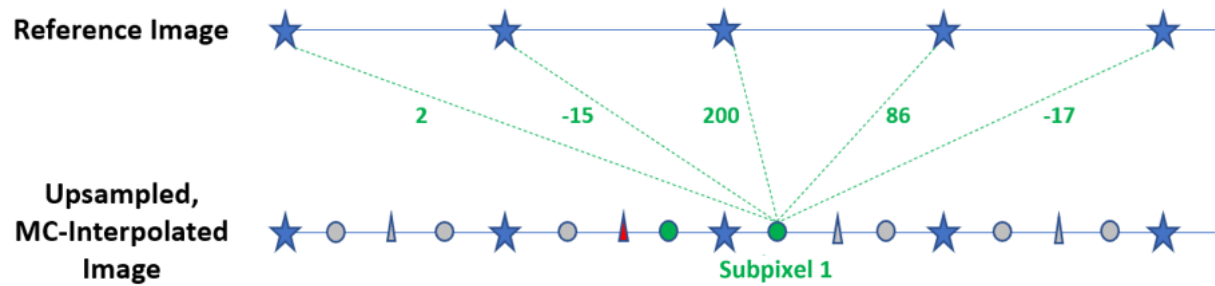
177. Following Subpixel 2, Subpixel 3 is then calculated by weighting all five reference pixel values with nonzero filter coefficients. Like Subpixel 2, Subpixel 3 still uses only one *additional* reference pixel to the left of the upsampled pixel location.



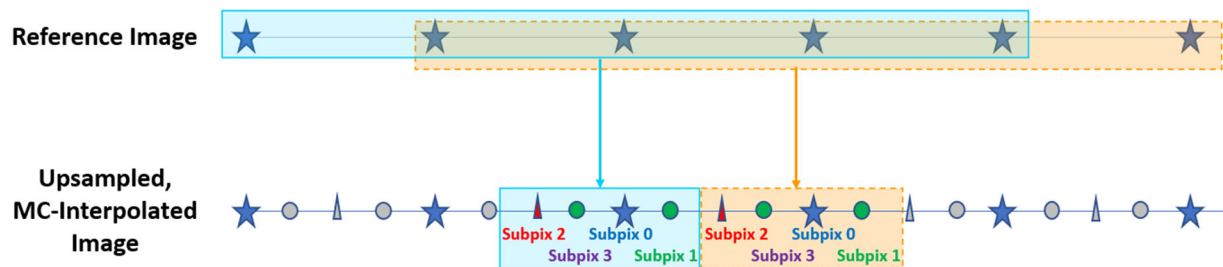
178. After Subpixel 3, Subpixel 0 is then calculated using two *additional* pixels to the left of the upsampled pixel location (the two left-most reference pixels shown in the figure below):



Finally, subpixel 1 follows the same pattern as shown below:



179. Summarizing, in Table 7C Sita discloses a single joint filter that performs the combined operations of 2:1 upsampling and half-pixel resolution motion compensation interpolation using a five-tap filter with four sets of filter coefficients for the four output subpixel locations. To further illustrate the operation of this filter for multiple cycles of the phases, I also provide an example illustrating two complete cycles of the filter phases below:



180. In my opinion, because Sita provides filter coefficients that are applied directly to the pixel values of the reference image to calculate the upsampled and motion-compensation-interpolated output pixel values without having to calculate or store any intermediate values (as would be output from the upsampling filter if the two filtering operations were implemented separately), Sita teaches a single filtering operation applied on a pixel such that $GF(s)=MCIF(SCF(s))$. Although my

examples above also walked through each subpixel calculation in a stepwise manner, Sita explains that by combining these operations, all four (for 2:1 upsampling) or six (for 3:1 upsampling) jointly filtered output pixel values can be determined in a single processor-clock cycle. Sita, ¶110 (“five parallel input ports may be provided to the upsampling block for each forward and backward lower and upper block, with five input pixels LWR(0)-LWR(4) for each clock transition for each reference block being combined with corresponding filter coefficients to provide one output pixel.”). In fact, this is the same process described by Interdigital’s expert during the ITC hearing initiated against Lenovo, except Interdigital’s expert used different weights than Sita in his example. Ex-1067, 525:11-527:21.

181. Moreover, although Sita states that the filtering process being described is for the “horizontal direction,” Sita also recognized that “[o]ne skilled in the art could easily extend the following discussion to motion translation in the vertical direction if desired.” Sita, ¶95. In my opinion, therefore, Sita’s description of horizontal filtering was merely representative of the filtering process, and a POSITA would have known how to apply it in the vertical direction as well, for the reasons I discussed previously. *See* Section IX.A.1.b (element 1[a]). For these reasons, in my opinion, Sita teaches a single filtering operation applied on a pixel such that $GF_v(s) = MCIF_v(SCF_v(s))$.

182. Based on these teachings, in my opinion, a POSITA would have been motivated to modify Nakagawa—both the decoding loop in the encoder, as well as a decoder implemented consistent with Nakagawa—to perform Sita’s joint resampling/motion-compensation-interpolation operations. A POSITA would have understood that Sita’s joint operations provide a compact and computationally efficient method in which pixels from a different-sized reference frame are retrieved to perform a combined upsampling and half-pixel-interpolation operation. *See* Sita, ¶108. Moreover, a POSITA would have recognized that Sita’s joint operations would reduce the use of computational resources in which lower memory bandwidth would be needed for fetching reference-frame pixels, while also streamlining the implementation of the resampling/motion-compensation-interpolation process. This would have been particularly important to Nakagawa, as one of the principal issues Nakagawa identifies in the prior art is the need to improve the latency (or time) to perform these operations. *See* Nakagawa, 1:32-2:46 (explaining a problem with the prior art is the delay in deciding to adapt the video-image resolution during encoding, and that “prior-art systems must execute each process faster than normal pipelined systems in order to finish the frame coding within a fixed cycle time.”).

183. In my opinion, a POSITA would have also had a reasonable expectation of success in implementing Sita’s joint operations in Nakagawa’s coding system because the modification would simply involve combining two mathematical

operations into one by selecting appropriate weights for pixel values to produce the same output as separate resampling and motion-compensation interpolation equations would have, and Sita even provides example sets of weights if 2:1 or 3:1 resampling were desired with half-pixel interpolation (although it would have been within the skill of a POSITA to calculate different weights for different resampling/interpolation scales). In my opinion, making this modification would involve little more than modifying some mathematical equations in computer code (and possibly determining different weights), which would have been well within the skill of a POSITA.

- f. 1[e] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”**

184. In my opinion, the combination of Nakagawa and Sita teaches element 1[e].

185. It is first worth noting that claim elements 1[d] and 1[e] are largely the same, except each are directed to one-dimensional filters: element 1[d] is directed to the vertical filter, and element 1[e] is directed to the horizontal filter. Both filters require the same two (joint) operations: resampling and motion-compensation interpolation.

186. As I explained previously, the combination of Nakagawa and Sita teaches the filter operations of element 1[d]. *See* Section IX.A.1.e. Unlike claim element 1[d], however, for claim element 1[e], a POSITA would not need to configure Sita’s filter operations from one direction (horizontal) to another direction (vertical) because Sita already expressly discloses the filter operations in the horizontal direction. *See id.*; Sita, ¶ 95. A POSITA therefore could have directly applied Sita’s teachings of a joint resampling/motion-compensation-interpolation operation in the horizontal direction to Nakagawa’s system. Therefore, in my opinion, the combination of Nakagawa and Sita teaches element 1[e] as well.

g. 1[f] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”

187. In my opinion, the combination of Nakagawa and Sita teaches element 1[f].

188. I had explained previously that Nakagawa discloses that reconstructed reference images are stored in a “frame memory,” and that when the reference image stored in the frame memory is a different size than the current frame, the reference image is resampled to match the size of the current frame. *See* Section IX.A.1.a (element 1[pre]); Nakagawa, 6:15-40.

189. However, Nakagawa also shows that no resampled version of the (reconstructed) reference image is stored in the frame memory. Specifically,

Nakagawa shows in Figure 2 that the “frame memory” provides the input to the resampling “CIF/QCIF converter,” which is external to the frame memory. Nakagawa, Fig. 2. Nakagawa also explains that images are resampled from the frame memory as needed. *See id.*, 6:15-34 (explaining that “[w]hen the CIF/QCIF selection controller 25 decided to use the CIF resolution ... [t]he CIF/QCIF converter 23 ... converts the retrieved picture to the CIF resolution if it has a QCIF resolution,” and “when the CIF/QCIF selection controller 25 has decided to use the QCIF resolution ... [t]he CIF/QCIF converter 23 retrieves the reconstructed picture of the previous frame from the frame memory 22 and converts it to the QCIF picture if it is of the CIF resolution.”). Nakagawa never discloses storing the converted pictures in the frame memory nor retrieving the converted pictures from the frame memory. Nakagawa’s Figure 2 is reproduced below with annotations:

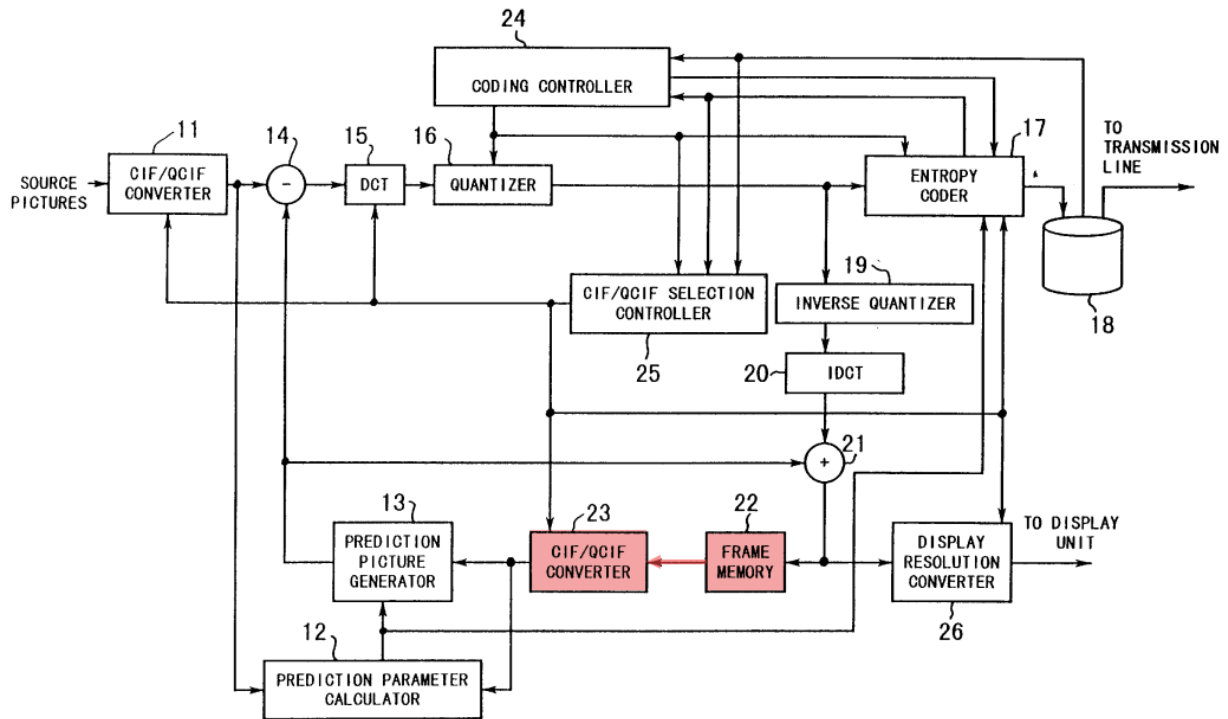
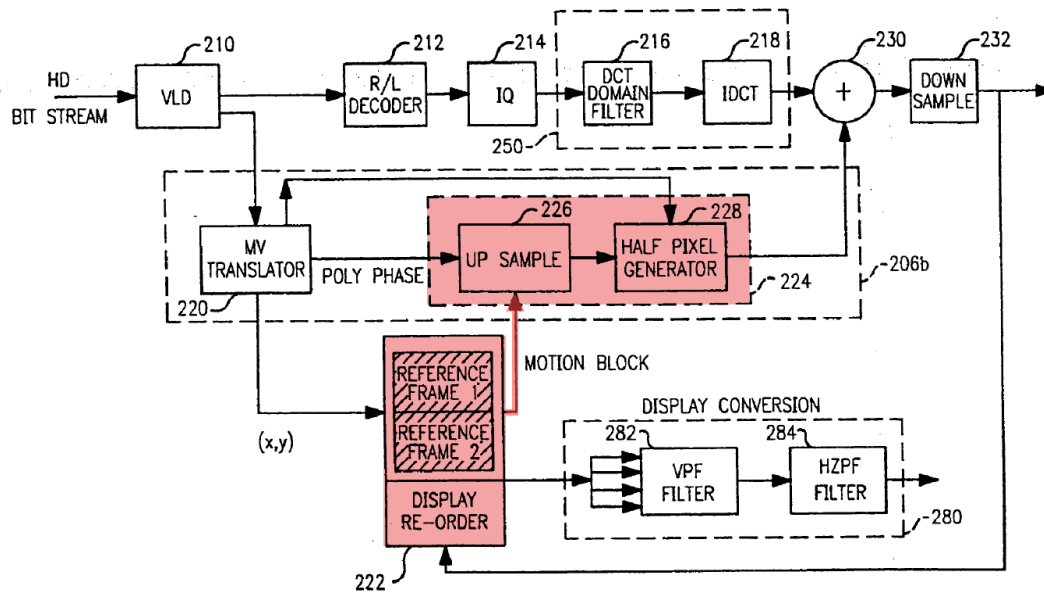


FIG. 2

In my opinion, because Nakagawa uses a “CIF/QCIF converter” external to the frame memory and never describes storing converted pictures in the frame memory after conversion, a POSITA would have understood that no resampled (i.e., converted) version of the reference image produced by the CIF/QCIF converter is stored in memory.

190. But a POSITA would have further appreciated that after implementing Sita’s joint resampling/motion-compensation-interpolation operations in Nakagawa’s system, no resampled version of the reference image would need to be stored in memory anyway. Sita taught a process to perform upsampling and motion-compensation interpolation in one operation, which eliminates the need to store

resampled versions of the reference images in memory. That is consistent with Sita's Figure 2B, which, like Nakagawa, shows reference frame memory (222) providing an input to filter (224), but filter (224) not providing any signal back to memory (222). An annotated version of Sita Figure 2B is provided below:



Moreover, as I explained previously, *see* Section IX.A.1.e (element 1[d]), Sita's filter always takes five reference-pixel values as inputs to generate the resampled, motion-compensation-interpolated subpixel values as outputs. Sita never describes a situation in this embodiment in which the filter takes already-resampled versions of the reference pixels as inputs. Accordingly, a POSITA would have understood that Sita also does not store resampled version of the reference pixels in memory.

191. And for the reasons I explained previously, *see* Section IX.A.1.a (element 1[pre]), if the decoder loop taught by Nakagawa and Sita would not need

to store a resampled version of the reconstructed reference frame in memory, a POSITA would have understood that a decoder configured accordingly would not do so either.

2. Claim 4

192. In my opinion, the combination of Nakagawa and Sita teaches or suggests all of the elements of claim 4.

193. As I explained previously in the context of claim 1, although Nakagawa primarily describes an encoder with an integrated decoder loop which at the very least would suggest the configuration of the decoder, Nakagawa's disclosure of the encoder is explicit. *See* Section IX.A.1.a (element 1[pre]).

- a. **4[pre] – “A coding method of a current block of a current image from a reference block of a reference image reconstructed at a different size from the size of said current image, said reconstructed reference image being stored in a decoded picture buffer comprising:”**

194. In my opinion, Nakagawa discloses or suggests element 4[pre]. *See* Section IX.A.1.a (element 1[pre]), IX.A.2 (claim 4, generally).

- b. 4[a] – “motion compensating said reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

195. In my opinion, the combination of Nakagawa and Sita teaches element 4[a]. *See* Section IX.A.1.b (element 1[a]).

- c. 4[b] – “calculating, for the current block, a residue block from said current block and from said motion compensated reference block, and”**

196. In my opinion, Nakagawa discloses or suggests element 4[b].

197. Specifically, Nakagawa discloses that “[t]he prediction error calculator 14 produces a prediction error signal by calculating differences between the current frame picture provided by the CIF/QCIF converter 11 and the prediction picture provided by the prediction picture generator 13.” Nakagawa, 5:13-17; *see also id.*, 5:2-12 (explaining how prediction picture generator applies motion vectors to a reference frame to produce the “predicted picture of the current frame”).

198. As I previously explained, in my opinion, a POSITA would have understood that Nakagawa’s “prediction error signal” is a “residue block,” and the “predicted picture” is a “motion compensated reference block,” as claimed. *See* Section IX.A.1.c (element 1[b]).

- d. 4[c] – “coding the residue block in a binary stream,”**

199. In my opinion, Nakagawa discloses or suggests element 4[c].

200. Specifically, Nakagawa discloses that the “prediction error signal” is sent through a “DCT processor” that produces “transform coefficients,” which are then quantized and sent to an “entropy coder” that “assigns different codes to the combinations of those received data ... which will be sent out to the transmission line in the form of bit stream at a constant transfer rate.” Nakagawa, 5:17-41. A POSITA would have understood that an entropy encoder is a way to reduce the number of bits required to represent a particular bitstream by assigning “codewords” of shorter bit sequences to more-frequent bitstream segments (like Morse code, where the most frequent letters used in the alphabet (“E” and “T,” according to Morse) are assigned the shortest code of one dot and one dash, respectively); *see, e.g.*, Ex-1014, ¶57 (“Entropy coding unit 37 entropy codes the quantized coefficients to even further reduce bit rate.”). Nakagawa also explains more generally that “the coding system generates a digital video bit stream,” Nakagawa, 3:60-63; *see also id.*, 5:38-41 (calling the output a “bit stream”), which, as I explained above, would be understood as a “binary stream” as claimed. *See* Section IX.A.1.a (element 1[pre]). Accordingly, in my opinion, a POSITA would have understood that Nakagawa discloses the coding of a residue block into a binary stream.

- e. **4[d] – “wherein said single vertical filter GF_v applied on a pixel s is such that GF_v(s)=MCIF_v(SCF_v(s)), where MCIF_v is a**

vertical motion compensation interpolation filter and SCFv is a vertical resampling filter, MCIFv and SCFv being applied jointly and”

201. In my opinion, the combination of Nakagawa and Sita teaches element 4[d]. *See* Section IX.A.1.e (element 1[d]).

f. 4[e] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”

202. In my opinion, the combination of Nakagawa and Sita teaches element 4[e]. *See* Section IX.A.1.f (element 1[e]).

g. 4[f] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”

203. In my opinion, Nakagawa discloses or suggests element 4[f]. *See* Section IX.A.1.g (element 1[f]).

3. Claim 7

a. 7[pre] – “A decoding device comprising at least one circuit configured to:”

204. In my opinion, Nakagawa discloses or suggests element 7[pre].

205. Specifically, Nakagawa discloses a “video coding apparatus for performing predictive coding of digital video input signals.” Nakagawa, Abstract. In my opinion, a POSITA would have understood that because Nakagawa is concerned

with encoding *digital* video signals, Nakagawa was intended to be implemented on a computer system with either specialized hardware, or computer processors and memory operating software. This would have met the clarifying construction Patent Owner agreed to in separate proceedings. *See* Section VIII (agreeing the term “circuit configured to ... requires pre-existing programming of hardware and software to perform the cited functionality”).

206. A POSITA would have further understood that a computer processor is itself a circuit made up of transistors and other components, and its coupling to memory forms an additional circuit because the data signals passing from the processor to memory are conveyed through electrical circuitry. In my opinion, a POSITA would have therefore understood a computer system to be “at least one circuit,” as claimed, more generally.

207. Furthermore, as I explained previously, a POSITA would have understood Nakagawa as disclosing or suggesting a decoder. *See* Section IX.A.1.a (element 1[pre]).

b. 7[a] – “access, from a decoded picture buffer, a reference image reconstructed at a size different from the size of a current image;”

208. In my opinion, Nakagawa discloses or suggests element 7[a]. *See* Section IX.A.1.a (element 1[pre]). Nakagawa also discloses that, as part of the decoder loop, reference images are accessed from a buffer or frame memory (e.g., a

decoded picture buffer) when reconstructing an image. *See, e.g.*, Nakagawa, 5:4-7 (“The prediction picture generator 13 receives ... a decoded image of the previous frame from the frame memory 22”).

- c. **7[b] – “motion compensate a reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

209. In my opinion, the combination of Nakagawa and Sita teaches element 7[b]. *See* Section IX.A.1.b (element 1[a]).

- d. **7[c] – “decode, for a current block of said current image, a residue block from a binary stream, and”**

210. In my opinion, Nakagawa discloses or suggests element 7[c]. *See* Section IX.A.1.c (element 1[b]).

- e. **7[d] – “reconstruct the current block from said residue block and from said motion compensated reference block,”**

211. In my opinion, Nakagawa discloses or suggests element 7[d]. *See* Section IX.A.1.d (element 1[c]).

- f. **7[e] – “wherein said single vertical filter GFv applied on a pixel s is such that $GFv(s)=MCIFv(SCFv(s))$, where MCIFv is a vertical motion compensation interpolation filter and SCFv is a vertical resampling filter, MCIFv and SCFv being applied jointly and”**

212. In my opinion, the combination of Nakagawa and Sita teaches element 7[e]. *See* Section IX.A.1.e (element 1[d]).

- g. 7[f] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”**

213. In my opinion, the combination of Nakagawa and Sita teaches element 7[f]. *See* Section IX.A.1.f (element 1[e]).

- h. 7[g] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”**

214. In my opinion, Nakagawa discloses or suggests element 7[g]. *See* Section IX.A.1.g (element 1[f]).

4. Claim 8

- a. 8[pre] – “A coding device comprising at least one circuit configured to:”**

215. In my opinion, Nakagawa discloses or suggests element 8[pre]. *See* Sections IX.A.1.a (element 1[pre]), IX.A.2 (claim 4, generally) (describing an encoder), IX.A.3.a (element 7[pre]) (describing a circuit).

- b. 8[a] – “access, from a decoded picture buffer, a reference image reconstructed at a size different from the size of a current image;”**

216. In my opinion, Nakagawa discloses or suggests element 8[a].
See Sections IX.A.1.a (element 1[pre]), IX.A.3.b (element 7[a]) (describing access).

- c. **8[b] – “motion compensate a reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

217. In my opinion, the combination of Nakagawa and Sita teaches element 8[b]. *See* Section IX.A.1.b (element 1[a]).

- d. **8[c] – “calculate, for a current block of said current image, a residue block from said current block and from said motion compensated reference block, and”**

218. In my opinion, Nakagawa discloses or suggests element 8[c]. *See* Section IX.A.2.c (element 4[b]).

- e. **8[d] – “code the residue block in a binary stream,”**

219. In my opinion, Nakagawa discloses or suggests element 8[d]. *See* Section IX.A.2.d (element 4[c]).

- f. **8[e] – “wherein said single vertical filter GFv applied on a pixel s is such that $GFv(s)=MCIFv(SCFv(s))$, where MCIFv is a vertical motion compensation interpolation filter and SCFv is a vertical resampling filter, MCIFv and SCFv being applied jointly and”**

220. In my opinion, the combination of Nakagawa and Sita teaches element 8[e]. *See* Section IX.A.1.e (element 1[d]).

- g. 8[f] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”**

221. In my opinion, the combination of Nakagawa and Sita teaches element 8[f]. *See* Section IX.A.1.f (element 1[e]).

- h. 8[g] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”**

222. In my opinion, Nakagawa discloses or suggests element 8[g]. *See* Section IX.A.1.g (element 1[f]).

B. Ground 2: Obviousness over Wiegand in view of Davies, and Fimoff

1. Claim 1

223. In my opinion, for the reasons discussed in detail below, Wiegand, Davies, and Fimoff together teach every element of claim 1.

- a. 1[pre] – “A decoding method of a binary stream to reconstruct a current block of a current image from a reference block of a reference image reconstructed at a different size from the size of said current image, said reconstructed reference image being stored in a decoded picture buffer comprising:”**

224. In my opinion, Wiegand and Davies teach element 1[pre].

225. Wiegand discloses the claimed “decoding method” by disclosing “[d]ecoding process” that “[o]utputs ... decoded samples of the current picture.” Wiegand, 84 (§8).

226. Wiegand also discloses the claimed decoding of a “binary stream,” although Wiegand uses slightly different language. In my opinion, a “binary stream” as used in the ’877 Patent simply refers to a stream of digital data. That is because digital data can only take one of two binary states, often depicted as true/false, 0/1, on/off, or positive/negative. Computer processors operate using transistors that recognize only these binary states: when voltage is applied to the transistor, the transistor switch is considered an “on” state, and when there is no voltage, the transistor is considered an “off” state. Likewise, hard drives also store binary data. The smallest unit of digital data—a single binary value—is called a “bit.” Thus, a “bit” can either be, e.g., a “0” or “1.”

227. With this understanding, Wiegand explains that the “[i]nputs to this process are NAL units,” which Wiegand defines as a “bitstream.” Wiegand, 85 (§8.1); *id.*, 16 (§3.11) (defining “bitstream” as “[a] sequence of bits that form the representation of coded pictures *Bitstream is a collective term used to refer either to a NAL unit stream or a byte stream.*”) (emphasis added); *see also id.*, 27 (§6.1) (“This subclause specifies the relationship between the NAL unit stream and byte stream, either of which are referred to as the bitstream.”).

228. A “NAL unit” refers to a “Network Abstraction Layer” unit, which is the way digital bitstreams are organized for encoding in both the MPEG-4/H.264/AVC and HEVC video-coding standards. These can be thought of as “packets” of data, like the discrete data packets used for internet transmissions. In other words, a video may be broken down into many smaller “NAL units” (or packets), each with a certain number of binary bits, which are then stored or transmitted for eventual decoding and playback.

229. Therefore, a NAL unit—or, more particularly, a NAL unit stream—is just a sequence of binary values (or bits) that are grouped together for storage or transmission purposes.

230. Wiegand also discloses that the decoding process for a binary stream “reconstruct[s] a current block of a current image from a reference block of a reference image,” as claimed.

231. When this element refers to a “block,” a POSITA would have understood that this refers to “block-based” coding, which was one of the most prevalent forms of encoding and decoding both images and video at the time. In fact, this was standard practice for JPEG images and H.264/MPEG-4 video. *See* Ex-1025, 5-6, 9, 11-12. Block-based coding is a process of breaking a picture down into sets of “macroblocks” and/or “blocks,” and encoding each macroblock and/or block based on a collection of parameters. *See generally*, Section IV.A.2.

232. However, because blocks in an image may be coded separately, the use of block-based coding can often result in what is called blocking artifacts when the image is reconstructed. An example image with blocking artifacts is provided below:



This image shows visible boundaries at the block edges because the reconstructed pixels within adjacent blocks were slightly different due to information loss in the encoding process, with the result that the decoded blocks do not “stitch together” or juxtapose seamlessly. To reduce the visibility of such blocking artifacts in reconstructed images, video decoders often use a de-blocking filter to smooth the boundaries between adjacent blocks in the decoded (i.e., reconstructed) image.

233. A POSITA would have further understood that by reciting “reconstruct a current block of a current image from a reference block of a reference image,” this claim element is referring to inter-prediction coding of the blocks in the video image. Inter-prediction coding is a process of using parts (e.g., “blocks”) of an already reconstructed reference image to decode the next, current image.

234. This is exactly what Wiegand discloses. Specifically, Wiegand discloses a decoding process called “inter prediction mode” that reconstructs a current block using blocks from reference images. Wiegand, 109 (§8.4); *id.*, 18 (§3.47) (defining “inter coding” as “[c]oding of a *block*, macroblock, slice, or picture that uses inter prediction.”) (emphasis added); *see also, e.g., id.*, 117 (showing an example inter prediction process using blocks). “Inter prediction” is defined as “a prediction derived from decoded samples of reference pictures other than the current decoded picture.” Wiegand, 18 (§3.48); *see also id.*, 19 (§3.88) (defining “reference picture” as “contain[ing] samples that may be used for inter prediction in the decoding process of subsequent pictures[.]”). In other words, Wiegand’s “inter prediction” mode involves reconstructing a current block of a current image from a reference block of a reference image.

235. The use of blocks in Wiegand’s inter-prediction mode is further confirmed by Wiegand’s “deblocking filter process” that is applied “after the completion of the picture construction.” Wiegand, 137 (§8.6.1). This process would have been understood by a POSITA as eliminating blocking artifacts that arise in reconstructed images, as I mentioned above, and thus confirmed Wiegand’s use of block-based coding for inter prediction.

236. In my opinion, therefore, a POSITA would have understood that Wiegand’s “inter prediction mode” refers to the reconstruction of a current block

using a block from a reference image because the current block is “predicted” based on a block from the reference image.

237. Wiegand also discloses that reconstructed reference images are stored in a decoded picture buffer. *See, e.g.,* Wiegand, 17 (§3.30) (defining “decoded picture buffer” as “[a] buffer holding decoded pictures for reference”), 221 (providing an example flow chart of a decoder picture buffer feeding reference frames to the “[d]ecoding process”). Moreover, just as with block-based coding, the use of a decoded picture buffer was also standard practice in the H.264/MPEG-4 standard to store reference images to later be used as predictions for reconstructing subsequent images. *See generally* Ex-1026; Ex-1023. In my opinion, therefore, a POSITA would have understood that Wiegand stores a reconstructed picture in the decoded picture buffer to serve as a reference for subsequent frames.

238. Although Wiegand discloses a decoding method that includes reconstructing a current block of a current image from a reference block of a reference image, Wiegand does not expressly disclose reconstructing a current block based on “a reference image ... at a different size from the size of said current image,” as recited in the preamble of the claim.

239. A POSITA would have understood that Wiegand was a working draft of the HEVC standard that was intended to be a living document. After Wiegand published the working draft of the HEVC standard, another participant in the HEVC

standardization process—Davies—proposed an amendment to Wiegand’s draft standard. Davies, 4 (§4) (proposing that “[v]arious modifications to the current Working Draft (WD) [3] are needed,” with reference number “[3]” citing to Wiegand).

240. Davies recognizes that “streaming [video] over packets networks frequently require[s] that the encoded stream adapt to changing network conditions.” Davies, 2 (§2.1). To accommodate changing network conditions, Davies explains that “[t]he encoder has two main tools at its disposal: bit rate reduction and changing the resolution.” *Id.* Davies proposes that the HEVC standard utilize the latter option and include a process by which the encoder could send different-sized frames to the decoder so that the system could “adapt seamlessly to adverse network conditions.” *Id.*, 1 (Abstract).

241. For instance, if a network was not congested by other internet traffic, the encoder could send full-sized (e.g., 4K) frames to the decoder. But if the network was congested, the encoder could send lower-resolution (e.g., High Definition, which has $\frac{1}{4}$ the number of pixels as 4K) frames instead because those lower-resolution frames would require much less data to transmit. If the encoder continued sending full-sized frames during times of network congestion, that could lead to noticeable delays or frames could get lost in transit. Reducing the resolution (and

therefore the amount of data required to transmit the video) would increase the likelihood that the frames would reach the decoder without undue delay.

242. A POSITA would have recognized that changing the resolution of video frames in response to network conditions, however, would have presented another problem. Specifically, because a decoder (and an encoder's decoding loop) in "inter prediction mode" uses past frames to reconstruct subsequent frames, if Davies's proposal were adopted, it would mean that a frame being used as reference to predict the current frame being reconstructed may have a size that is different from that of the current frame, e.g., when the encoder switches resolution in the middle of a video.

243. To that end, Davies explains how past video encoders such as "AVC" (another name for the MPEG-4/H.264 codec) would essentially avoid this issue by sending an "IDR" frame—a special type of fully encoded "I" frame intended to fully "refresh" or reset the decoder. *Id.*, 2 (§ 2.1). However, Davies recognizes that "[a]n IDR frame at reasonable quality will be much larger than an Inter [prediction] picture, and will be correspondingly more complex to decode," and "can also break low-latency buffer conditions," all of which "gives a poor user experience." *Id.*

244. To avoid sending a resource-intensive IDR frame, Davies proposes further modifying Wiegand to allow frames to be "predicted across resolutions by re-scaling reference pictures," *id.*, 1 (Abstract), specifically by using "separable

down- and up-scaling filters ... for prediction,” *id.*, 4 (§3.2). In other words, Davies proposes that the decoder resample reference frames to change their size, thereby ensuring they match the size of the frame being predicted without needing to send an undesirable IDR frame. *Id.*, 1 (Abstract).

245. For instance, applying Davies’s proposal, if a video switches from a lower-resolution (HD) to a higher-resolution (4K) because more network bandwidth has become available, at the point where the video switches resolutions, the reference frames that had already been received and reconstructed by the decoder would be smaller (HD) than the incoming frame (4K) that must be predicted. To account for this, Davies proposes that the decoder “upscales” the lower-resolution references (e.g., from HD to 4K) so that the size of the reference image would match that of the frame intended to be predicted. This upscaling would be achieved using an interpolation process, which I described previously. *See* Section IV.A.3. Similarly, when the reference frame has a higher resolution (e.g., 4K) than the current frame being predicted (e.g., HD), the reverse process would be applied: the reference image would be “downsampled,” which effectively reduces the number of pixels in the image by deleting and/or modifying pixel values, which I also described previously. *See id.*

246. In my opinion, a POSITA would have been motivated to modify Wiegand’s inter prediction decoding process to allow the decoder to receive

different-sized frames, and then perform a resampling operation to ensure the size of a particular reference frame matches the size of the frame being predicted, as taught by Davies. A POSITA would have been motivated to make this change not only because Davies expressly proposes modifying Wiegand to accommodate different-sized frames, *see* Davies, 4 (§ 4), 11, but also because following Davies's proposal would have allowed video quality to be maximized even as video frames are encoded, transmitted, and decoded under adverse or variable network conditions. Furthermore, a POSITA would have recognized that modifying Wiegand to accommodate different-sized frames of Davies would mean that the resulting arrangement could avoid sending bandwidth-intensive IDR frames to switch video resolutions.

247. In my opinion, a POSITA would have had a reasonable expectation of success in supplementing the teachings of Wiegand with the teachings of Davies because Davies had already provided a description of the changes to be made to Wiegand (*see, e.g.*, Davies, 4-5 (§4.1) (describing modifications to bitstream syntax and semantics), *id.*, 6 (§4.2) (describing a picture parameter set that includes a picture resolution indicator), *id.*, 7 (§4.4.1) (describing items to modify), *id.*, 5 (describing simulation scenarios that included multiple sets of parameters to enable different configurations, and also describing performance metrics for the simulation scenarios)). Modifying Wiegand using the teachings of Davies would have involved

computer coding to update parameter sets using the definitions provided in Davies, which would have been well within the skill of a POSITA.

- b. 1[a] – “motion compensating said reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

248. In my opinion, the combination of Wiegand, Davies, and Fimoff teaches element 1[a].

249. To first understand the scope of the claim, although this claim element uses the abbreviations “GFH” and GFv,” in my opinion, a POSITA would have understood that these abbreviations refer to the set of “single filters” described in the specification: one applied in the horizontal (GFH) direction, and another applied in the vertical (GFv) direction. *See, e.g.*, ’877 Patent, 5:58-62 (“[T]he horizontal filters $MCIF_H$ and SCF_H are grouped into a single filter GF_H ,” and same for vertical). The abbreviations themselves therefore appear to do little to no work in the claim, which already requires “a single horizontal filter” and “a single vertical filter.” Instead, the abbreviations appear to just be references to the respective vertical and horizontal filters as described in the specification.

250. With this understanding, as I discussed previously, combining Wiegand and Davies would have produced a decoding process that adapts to changing network conditions by receiving different-sized frames, and then resizing frames

used as references as needed to match the size of the current frame being predicted.
See Section IX.B.1.a (element 1[pre]).

251. In my opinion, however, upon making this combination, a POSITA would have been aware that resizing reference frames on demand may overload the buffer (memory) used to store the reference frames. Specifically, Davies proposes modifying the “[r]eference picture selection process” that Wiegand had previously proposed would be the “same as AVC” (or MPEG-4/H.264). *See* Davies, 7 (§4.4.1.2) (modifying only §8.4.2.2.1 of Wiegand). As Wiegand explains, however, the reference picture selection process is a different step from the prediction process that reconstructs current frames from the reference frames that are selected. Wiegand, 124 (§8.4.2.2.1 describing the “[r]eference picture selection process”), 124-30 (§8.4.2.2.2) (describing prediction process through motion compensation and interpolation). Therefore, a POSITA would have understood that by modifying Wiegand with Davies, the resized reference images may need to be stored in Wiegand’s decoded picture buffer so that they could be used for later prediction because the two steps were described by Wiegand as different processes.

252. A POSITA would have been well aware of the limited capacity of decoded picture buffers, which was a focal point in past video coding standards. *See, e.g.,* Ex-1025, 4 (“H.264/MPEG-4 part 10 Video coding standard requires that the bit stream to be transmitted should be decodable by a Hypothetical Reference

Decoder (HRD) without an underflow or overflow of the reference buffer.”), 20 (noting need for memory management of the buffer); Ex-1035, 1 (noting issues with the decoder buffer and proposing a rate-control mechanism to prevent buffer overflow). The buffer had even been identified by participants in the JCT-VC as a point-of-concern before Wiegand had issued the first working draft of the HEVC standard. *See, e.g.*, Ex-1063, 2 (“The purpose of this algorithm is to reduce external memory access bandwidth and the reference frame buffer memory size in video coding.”).

253. In fact, another participant in the JCT-VC working group (Misra) even recognized a potential buffer-overflow issue if *Davies’s* proposal were adopted because it may require storing multiple copies of the same frame—each at different resolutions after Davies’s rescaling operation—in the buffer. *See* Ex-1013, 1 (“If a stream switches to a lower resolution without an IDR [as Davies proposed], ... then the decoded picture buffer may contain pictures having different resolutions.”). Although Misra appeared to agree that offering a resizing operation was still worthwhile, to avoid any buffer overflow, Misra proposed that Davies (and, thus Wiegand’s draft standard) could be further modified to have the encoder tell the decoder which sized reference frames should be stored in the buffer to prevent overflow. *Id.* But even Misra did not expressly describe the exact process by which images could be rescaled and predicted based on different-sized reference images

without storing the rescaled reference images in the buffer; Misra just says there could be a blanket prohibition on storing resized images.

254. Therefore, even though the combination of Wiegand and Davies would have provided all the pieces required to predict a current frame based on a different-sized reference frame, the combination would have posed an additional problem of possible buffer overflow. And even turning to Misra would not have solved exactly how frames could be rescaled and used as references *without* storing the rescaled reference images in the buffer. A POSITA therefore would have been in search of another solution.

255. A POSITA would have recognized that controlling buffer size when decoding a current frame based on a different-sized reference frame had already been solved for a similar application in an earlier video standard. For instance, as I noted previously, television networks began switching from broadcasting standard-definition television signals to high-definition television signals around the turn of the millennium, Ex-1034, even though the adoption of high-definition televisions was not yet widespread. One exemplary prior-art reference—Fimoff—sought to ensure that legacy, standard-definition televisions could still receive and play back these high-definition video signals. Fimoff, 3:37-47. In my opinion, a POSITA would have been well aware of Fimoff and the solution it proposes because of the

massive impact the switch from standard-definition to high-definition television signals had on the field, and the need to offer continued support for legacy systems.

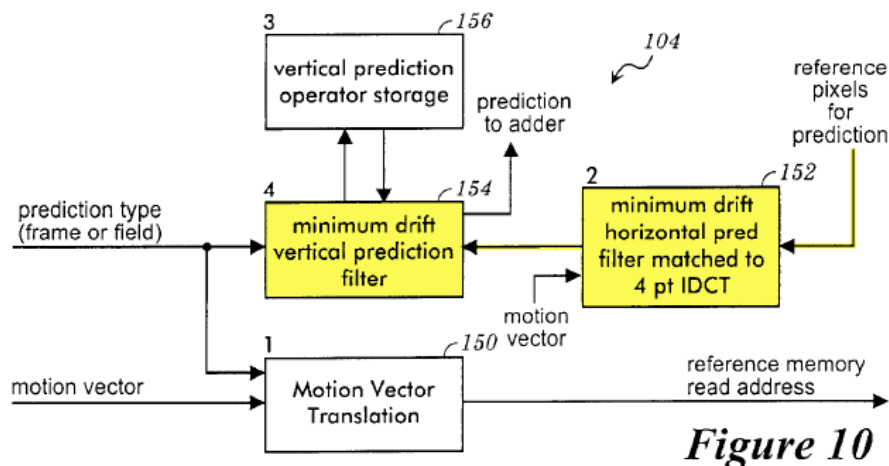
256. To provide continued support for legacy systems, Fimoff proposes performing a series of filtering operations in a single pass that would allow for predictive decoding across different resolutions, while ultimately converting the resolution of high-definition video content to standard definition. Fimoff, Abstract, 3:37-47, 21:15-17, 24:24-28.

257. As part of that process, Fimoff discloses a decoder that would receive high-definition frames, which are then downsampled and stored in memory to serve as reference frames. *See, e.g., id.*, 2:39-45 (describing treatment of I frames in MPEG-2 generally), 2:66-3:3 (describing creation and storage of reference frames); *see also id.*, 7:66-8:3 (describing a downsampling module 102 that operates on the incoming signal), Fig. 5 (showing an example down-converting decoder).

258. When a previously reconstructed frame is needed as a reference, Fimoff explains that the decoder converts the full-resolution motion vectors for the current frame received from the encoder (which were generated based on a high-definition reference frame at the encoder), to low-resolution motion vectors. *Id.*, 30:11-24. The conversion process occurs so that the decoder can identify which pixels in the reference image are impacted by a particular motion vector. Then, Fimoff retrieves the relevant set of impacted pixels of the reference image from memory and

upsamples them to convert the impacted portion of the standard-definition reference image back into high-definition. *Id.*, 18:6-19. Accordingly, the full-resolution motion vectors provided by the encoder can be accurately matched to the now-high-definition reference pixels at the decoder. *See id.*, 19:22-31, Fig. 7. This upsampling also enables subpixel interpolation when accounting for half-pixel resolution motion vectors permitted by MPEG-2.

259. Figure 10, which is reproduced below, shows the signal flow through the combined resampling/motion-compensation-interpolation filter, which comprises separate filters for the horizontal and vertical direction:



Fimoff explains how a single vertical filter would be applied successively on the columns of pixels of the reference block. *See, e.g.*, Fimoff, 30:31-34 (explaining the “vertical filter operator [PL] is ... applied to each column of pixels as they are received from the horizontal prediction filter.”). Although Fimoff does not say

exactly how the horizontal filter operates (because Fimoff was primarily concerned with the vertical filter), a POSITA would have understood that Fimoff's horizontal filter is applied successively to lines of pixels as well because this was standard practice of horizontal filters: they are applied to lines (or rows) of pixels in the horizontal direction. *See, e.g.*, Sita, ¶123 (describing another coding system similar to Fimoff's, noting "[i]t will be apparent to one skilled in the art ... that multidimensional filtering may be accomplished by filtering the rows and then filtering the columns of the block matrix."); Ex-1049, ¶30 ("applying a horizontal deblocking filter to each set 630 of eight horizontally arranged pixels"), Fig. 6C.

260. After performing a motion-compensation interpolation operation on the upsampled reference pixels, the reference pixels are then again downsampled to standard-definition resolution for display on older, standard-definition televisions. Fimoff, 20:25-36, 20:60-21:14, 21:52-55. Fimoff explains that, through this process, "prediction upsampling, linear interpolation [for motion compensation], and prediction downsampling," can occur in "a single operation." *Id.*, 21:15-17, 21:55-60 (same), Figure 5 (showing the single operation of the motion compensator (104) performed on either frame pictures or field pictures). In other words, Fimoff's motion compensator (104) performs both resizing and motion-compensation-

interpolation operations using a single horizontal filter and a single vertical filter, and stores only one size of a resulting reference image in memory.⁵

261. According to Fimoff, performing these filtering operations in a single pass would require less “reference picture memory,” Fimoff, Abstract; *see also id.*, Title, because only the low-resolution versions of the video frames would need to be stored in memory, *id.*, 30:11-24. Although not stated expressly, in my opinion, a POSITA would have also understood that Fimoff’s memory savings are derived from Fimoff’s ability to resize and interpolate reference frames to predict current frames without having to store those resized reference frames in memory (buffer).

262. In my opinion, a POSITA therefore would have been motivated to modify Wiegand’s and Davies’s *separate* resampling and motion-compensation-interpolation filters to be applied together as a single filter based on Fimoff—as described in further detail below, *see* Sections IX.B.1.e, IX.B.1.f (elements 1[d] and

⁵ For the reasons I explain in further detail below, *see* Section IX.B.1.e, although Fimoff describes a final downsampling procedure to convert the video images to standard-definition, a POSITA would have known that this final downsampling procedure would be unnecessary for Wiegand’s and Davies’s combined system.

1[e])—to avoid overwhelming the buffer with multiple versions of the same reference frame at different sizes.

263. A POSITA would have turned to Fimoff because, as with Davies, Fimoff was concerned with resizing video images based on resource availability, and both concerned predicting current frames from different-sized references. Furthermore, by using Fimoff's process, a POSITA would have also recognized that computational resources would be conserved by avoiding having to route the decoded signal through separate resampling and motion-compensation-interpolation filter operations and sending reference frames to/from memory multiple times.

264. In my opinion, a POSITA would have also had a reasonable expectation of success in making this modification because Fimoff already provided a description (including equations) sufficient to implement a combined resizing-interpolation filter operation. Moreover, this modification involves little more than combining separate, known mathematical processes for resampling and motion-compensation interpolation into a single mathematical process. To implement the combined operations in Wiegand's and Davies's combined system, a POSITA would have needed familiarity with computer programming to implement the resampling and motion-compensation interpolation as a single mathematical operation, which I would consider to be within the skills of a POSITA.

265. Moreover, a POSITA would have known that Fimoff was not a one-off solution to this problem, and that others had proposed similar filtering operations that avoided storing multiple versions of a reference frame at different sizes in the buffer. As I described previously, Sita, for instance, also sought to decode high-definition television signals to display on standard-definition televisions, *see, e.g.*, Sita, ¶2, just like Fimoff. As part of the process, Sita also describes a combined upsampling and motion-compensation-interpolation filter (224), *id.*, ¶80, and also avoided storing resampled versions of the already-reconstructed reference images in a buffer, *see, e.g.*, Sita, Fig. 2B (showing upsample/interpolation filter 224) that is applied without storing the upsampled image in memory), ¶52 (noting goal of minimizing memory use); *see also id.*, ¶¶78, 82, just like Fimoff. A POSITA would have therefore understood that Fimoff's use of a single filter was a well-known option in the art.

c. 1[b] – “decoding, for the current block, a residue block, and”

266. In my opinion, Wiegand discloses element 1[b].

267. First, a POSITA would have understood generally that a “residual” or “residue” block is nothing but a set of data that captures any remaining errors or differences between a reference block and the current block being predicted, as I explained previously. The current block is reconstructed by taking an existing,

matching block from a reference image, applying motion vectors (for motion compensation), and adding back the “residual” or “residue” determined by the encoder that accounts for any other differences. Fimoff, for instance, describes this as the standard practice of the older MPEG-2 codec. Specifically, at the encoder:

[t]he motion estimator 14 reads [a] matching macroblock (known as a predicted macroblock) out of the reference picture memory 16 and sends it to the subtractor 12 which subtracts it, on a pixel by pixel basis, from the new macroblock entering the MPEG-2 encoder 10. The output of the subtractor 12 is an error, *or residual*, that represents the difference between the predicted macroblock and the new macroblock being encoded.

See Fimoff, 1:57-64 (emphasis added). After this residual is encoded into a data signal, at the decoder:

the residual from the quantization block 20 [at the encoder] is inverse quantized... [E]ach motion vector is translated by the motion compensator 34 to a memory address in order to read a particular macroblock (predicted macroblock) out of a reference picture memory 42 which contains previously stored reference pictures. An adder 44 adds this predicted macroblock to the residual provided by the IDCT block 38 in order to form reconstructed pixel data.

Id., 2:20-27, 2:60-66.

268. Similarly, Wiegand explains that a “residue” or “residual” is the “difference between a prediction of a sample or data element and its decoded value.” Wiegand, 20 (§3.95). Wiegand then discloses that a “[d]ecoding process for coding units coded in inter prediction mode consists of ... [t]he decoding process for the residual signal of coding units coded in inter prediction mode[.]” *Id.*, 109 (§8.4). Wiegand further provides details and examples regarding how the residue is decoded. *See id.*, 130-33 (§8.4.3).

269. For these reasons, in my opinion, a POSITA would have appreciated that Wiegand teaches the use of residuals in image decoding, and specifically decoding a residual for the current block. A POSITA would have understood this even though Wiegand contained a note that the final reconstruction process would be “[r]evisit[ed] after writing picture reconstruction process” because Wiegand’s note concerned only the detailed implementation of the color components for purposes of the HEVC standard.

270. Moreover, a POSITA would have also appreciated that the ’877 Patent was not reciting any new teachings as to the use of a residual, but was just following standard practice of using a residual to reconstruct the current block, as had long been done before. Thus, Wiegand’s lack of particular detail would not have posed any sort of problem to a POSITA, given that decoding residuals and using them for reconstruction had been done for decades.

d. 1[c] – “reconstructing the current block from said residue block and from said motion compensated reference block,”

271. In my opinion, the combination of Wiegand, Davies, and Fimoff teaches element 1[c].

272. As I explained above, the combination of Wiegand, Davies, and Fimoff teaches the generation of a predicted, motion compensated reference block by resampling and motion compensating a block from a different-sized reference image. *See* Sections IX.B.1.a, IX.B.1.b (element 1[pre] and 1[a]). Furthermore, as I also explained above, Wiegand discloses decoding a residual block of the current block. *See* Section IX.B.1.c (element 1[b]).

273. Wiegand further discloses that, to reconstruct an image, first the predicted, motion compensated reference block is decoded (pursuant to §§8.4.1 and 8.4.2). Wiegand, 109 (§8.4, step 1); *see also id.*, 110 (§8.4.1 directing to follow §8.4.2); *see also, generally, id.*, 111-130 (describing calculation of motion compensation). Next, the residue of the block is decoded (pursuant to §8.4.3). *Id.* Finally, a picture reconstruction process is followed. *Id.* (step 4).

274. As I noted above, although Wiegand does not describe the exact reconstruction process, the reconstruction of a current block based on a motion-compensated reference image and residual block was standard practice at the time. *See* Section IX.B.1.c (element 1[b]). In my opinion, a POSITA therefore would have

understood that Wiegand would follow this same basic process to reconstruct frames as well.

- e. **1[d] – “wherein said single vertical filter GF_v applied on a pixel s is such that $GF_v(s)=MCIF_v(SCF_v(s))$, where MCIF_v is a vertical motion compensation interpolation filter and SCF_v is a vertical resampling filter, MCIF_v and SCF_v being applied jointly and”**

275. In my opinion, the combination of Wiegand, Davies, and Fimoff teaches element 1[d].

276. Applying the understanding of the claim set forth above, *see* Section IX.A.1.e (Ground 1, element 1[d]), Fimoff teaches a single vertical filter (154 in Fig. 10) comprising a vertical motion compensation interpolation filter (MCIF_v) and a vertical resampling filter (SCF_v) with the resampling filter applied first as part of a joint operation. Fimoff, 20:1-24 (resampling filter [u]; field prediction for field pictures); *id.*, 20:60-64 (“up sampling must be followed by $\frac{1}{2}$ pixel linear interpolation” for vmvtype=0 odd-numbered vertical motion vectors), 21:15-17 (“Vertical prediction up sampling, linear interpolation, and prediction down sampling can be combined into a single operation” for vmvtype=0), *generally id.*, 20:31-21:27 (filtering vmvtype=0, generally); *see also id.*, 21:52-54 (“up sampling must be followed by $\frac{1}{2}$ pixel linear interpolation” for vmvtype=1 odd-numbered vertical motion vectors), 21:55-60 (“Vertical prediction up sampling, linear interpolation, and prediction down sampling can be combined into a single

operation” for $\text{vmvtype}=1$), *generally id.*, 21:28-67 (filtering $\text{vmvtype}=1$, generally); *see also id.*, 22:47-65 (resampling filter [u] (SCFv); field prediction for frame pictures); 23:5-24:5 (disclosing combined filter for $\text{vmvtype}=0$); 24:6-55 (disclosing combined filter for $\text{vmvtype}=1$), 30:25-38; ’877 Patent, 5:57-67 (“According to the invention, the horizontal filters MCIF_H and SCF_H are grouped into a single filter GF_H applied on the lines of a pixel block” using, e.g., equation $\text{GF}_H = \text{MCIF}_H \circ \text{SCF}_H$ (with the same notation used for vertical filter “V”), “where $(f \circ h)(x) = f(h(x))$.”).

277. Although the claim requires that these two filters be “applied jointly,” in my opinion, a POSITA would have understood that Fimoff’s filters are “applied jointly” based on how this term was used during prosecution of the ’877 Patent. Specifically, although the phrase “applied jointly” does not appear in the specification (only the claims), *see* ’877 Patent, during prosecution of the ’877 Patent, the applicant used the phrase “applied jointly” to mean “applied using a single filter.” Ex-1002, 281. The applicant also distinguished the prior art because it applied a resampling and motion compensation filter sequentially using two different modules. *Id.* Therefore, I understand “applied jointly” to require a single operation to perform both resampling and motion compensation. And, in my opinion, Fimoff teaches this feature. *See* Fimoff, 21:15-65.

278. Although Fimoff taught the use of an additional downsampling filter to convert the predicted frames from a high-definition to a standard-definition format

so the images could be displayed on standard-definition televisions, Fimoff, 21:10-27, in my opinion a POSITA would have further understood that this additional downsampling operation would be unnecessary in the combination of Wiegand and Davies. Specifically, Wiegand and Davies together concerned adaptively resizing different frames in a video sequence based on the capacity of the network to transmit those videos. In other words, Wiegand and Davies were not concerned with consistently outputting standard-definition video, like Fimoff was. When looking to Fimoff to further improve Wiegand's and Davies's combined decoder, a POSITA would have understood that resampling reference images would be required only to ensure that the motion vectors of the current frame would align with the correct pixels in the reference frame if the two were different sizes. A POSITA therefore would have understood that Fimoff's final downsampling operation would be unnecessary in Wiegand's and Davies's combined system. A POSITA would have further appreciated that Fimoff's final downsampling operation could easily be deactivated by, for example, setting the matrix $[d0]$ in Fimoff's Equation (66) to an identity matrix (instead of the matrix given in Fimoff's Equation (60)) and setting the matrix $[d1]$ in Fimoff's Equation (70) to an identity matrix (instead of the matrix given in Fimoff's Equation (68)) – a modification that would have been well within the technical capabilities of a POSITA. A POSITA would have been motivated to implement this modification because doing so would not only deactivate Fimoff's

unnecessary final downsampling operation, but would also simplify the design of the final combined filters given in Fimoff's Equations (66) and (72). *See* Ex-1011, 21:15-28, 21:52-67.

279. Finally, as I described above, a POSITA would have been motivated to modify Wiegand and Davies to apply their separate resampling and motion-compensation filters together as part of a single filter, as taught by Fimoff. *See* Section IX.B.1.b (element 1[a]).

- f. 1[e] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”**

280. In my opinion, the combination of Wiegand, Davies, and Fimoff teaches element 1[e].

281. As I discussed previously, Fimoff describes the claimed single filter in the vertical direction. *See* Section IX.B.1.e (element 1[d]). Fimoff also describes that “[h]orizontal prediction filtering occurs in the horizontal prediction filter 152,” Fimoff, 30:22-25, and shows a “horizontal prediction filter” (152) in Figure 10, which is reproduced below:

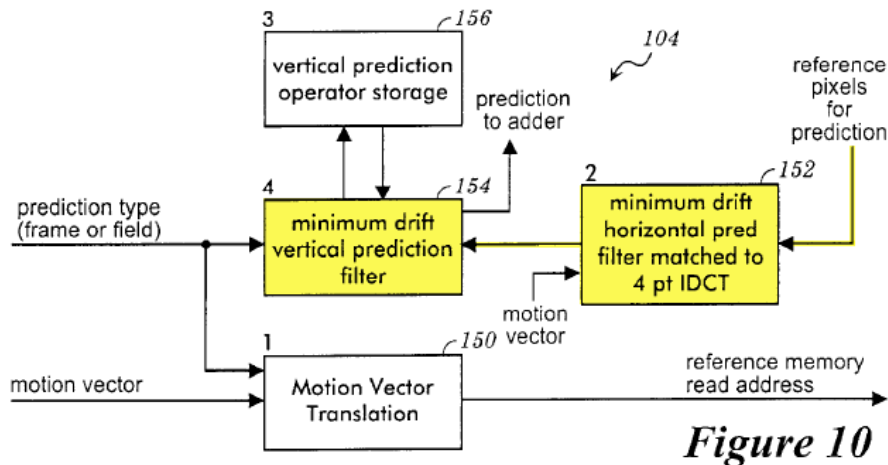


Figure 10

However, Fimoff does not expressly describe the exact composition and operation of the horizontal filter in the same level of detail as the vertical filter.

282. In my opinion, Fimoff did not describe the horizontal filter because it did not need to. Specifically, Fimoff appears to focus on the vertical filter because Fimoff was concerned in part with accounting for issues that arise when trying to downsample a specific type of “interlaced” video, and this concern warranted further consideration in the vertical direction due to the nature of these types of videos, which could include field-coded, frame-coded, or mixed-type pictures. *See, e.g.,* Fimoff, Abstract, 5:36-41, 30:39-54. I discussed Fimoff’s focus on interlaced video in my summary of Fimoff above. *See* Section VI.E.

283. Fimoff’s horizontal filter, however, did not require similarly detailed consideration because horizontal lines of pixels in field-coded and frame-coded pictures could be processed in a similar manner. *See* Fimoff, 30:12-24. Lines of

pixels in interlaced video remain the same regardless of whether they are field-coded or frame-coded; there are no gaps between pixels in the horizontal direction like there are for fields in the vertical direction. *See* Fimoff, 16:43-47 (“[a]ll blocks undergo the same horizontal processing (the four-point IDCT), but for vertical processing, the [QL] operator is used when necessary to effectively convert the frame coding to field coding ...”).

284. Accordingly, it would have been much more straightforward to implement Fimoff’s horizontal filtering because many of the particular considerations to filter interlaced video in the vertical direction would be irrelevant in the horizontal direction.

285. Because Wiegand and Davies already taught a process of resampling a reference image followed by a motion-compensation operation, *see* Section IX.B.1.b (element 1[a]), the only question is whether a POSITA would have applied the two together in both the vertical and horizontal directions based on Fimoff’s teachings. While I already addressed applying those filters together in the vertical direction, *see* Section IX.B.1.e (element 1[d]), in my opinion a POSITA would have also applied them together in the horizontal direction as well.

286. A POSITA would have appreciated that, even though Fimoff did not expressly provide many of the details of the horizontal filter, Fimoff’s use of a combined resampling and motion-compensation-interpolation filter would not be

limited to the vertical direction. Specifically, a POSITA would have appreciated that the standard-definition images Fimoff stored in reference memory would need to be upsampled not only in the vertical direction, but also in the horizontal direction as well to maintain the aspect ratio of the image. Otherwise, the image content would be visually distorted in one direction relative to the other when the images were subsequently resized for prediction.

287. For instance, a POSITA would have understood that the high-definition images received by Fimoff's decoder could have up to 1920 pixels in each horizontal row because that is the standard horizontal size of high-definition images. But a POSITA also would have understood that standard-definition NTSC format referred to by Fimoff generally had 720 horizontal pixels in each row. *See* Fimoff, 3:37-47. In Fimoff's system, therefore, the standard-definition images in reference memory would need to be resampled in the vertical direction and in the horizontal direction to match the number of high-definition pixels received by the decoder. Accordingly, a POSITA would have understood that Fimoff would require a resampling filter in both the vertical and horizontal directions in order to resize reference images in both directions and maintain their aspect ratio.

288. Furthermore, a POSITA would have appreciated that motion compensation does not occur in one dimension either, and that objects in an image can move both horizontally and vertically relative to reference images. Thus, a

POSITA would have understood that Fimoff's horizontal filter would have needed a motion-compensation-interpolation filter too to account for any horizontal components of the motion vectors. Without a horizontal motion-compensation-interpolation filter, only the vertical components of the motion vectors would be compensated.

289. Although a POSITA would have understood that Fimoff's horizontal filter would apply at least a resampling filter and motion-compensation-interpolation filter in the horizontal direction as well as the vertical direction, a POSITA *also* would have known to apply those two filters together as part of a single operation, just like Fimoff expressly does in the vertical direction.

290. Indeed, other prior art had recognized that resampling/motion-compensation-interpolation filters in one direction could be routinely adapted to apply in a different direction. Sita, for instance, chose to describe the details of the upsampling/motion-compensation-interpolation filter in the horizontal direction, and then expressly noted that “[o]ne skilled in the art could easily extend the following discussion to motion translation in the vertical direction if desired.” Sita, ¶ 95.

291. In other words, Sita confirms that a POSITA would know how to use a description of an upsampling/motion-compensation-interpolation filter described for

one direction as a guide to implement the filter in another direction with minimal effort.

292. Other prior art had further recognized the applicability of combined-operation filters to different directions. Specifically, WO 1990/00780 (“Dolazza”), published in January 1990, disclosed another coding apparatus for simultaneously filtering and resampling image data. Ex-1016, 1 (Abstract). As part of the invention, Dolazza recognized that “identical, serially coupled, programmable processing modules for simultaneously filtering and resampling image pixel data” could be separately assigned for the vertical and horizontal direction. *Id.*, 19:22-33; *see also*, *id.*, 3:22-28.

293. Similarly, U.S. Patent Publication No. 2008/0089417 (“Bao”) took this a step further and noted that motion-compensation filters could be combined with other types of filters (such as low-pass, smoothing, or bilinear filters) in a single pass, where the horizontal and vertical filters could be configured to have the same or different filters depending on what goal the filter was intended to achieve. Ex-1014, ¶¶39-40. A POSITA would have understood that these filters were treated as modular units that could be combined and applied in different directions depending on the particular needs of the specific situation.

294. In sum, a POSITA would have been motivated to modify Wiegand and Davies to combine the resampling and motion-compensation-interpolation processes

into a single operation, *see* Section IX.B.1.b (element 1[a]), and apply that operation in each direction as part of a horizontal and vertical filter, as taught (for the vertical) or suggested (for the horizontal) by Fimoff.

- g. 1[f] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”**

295. In my opinion, the combination of Wiegand, Davies, and Fimoff teaches element 1[f].

296. As I explained previously, Wiegand teaches the use of a “decoded picture buffer” to store reference frames. Section IX.B.1.a (element 1[pre]). However, as I also explained previously, a POSITA would have been well aware of the limited memory availability in the buffer. *See* Section IX.B.1.b (element 1[a]). It was for this reason that a POSITA would have been motivated to apply Fimoff’s teachings to the combination of Wiegand and Davies to *avoid* overloading the buffer by eliminating the need to store resized reference frames during the combined resampling and motion-compensation-interpolation filtering. *See id.*

297. Specifically, although Fimoff does not use the term “decoded picture buffer,” Fimoff discloses a decoder that includes “memory 106.” Fimoff, 30:15-22. A POSITA would have understood these to be one-and-the-same because Fimoff’s “memory” serves the same purpose as the claimed “decoded picture buffer”: to store reference images. Fimoff, Fig. 5 (showing memory 106 storing “[r]educed resolution

reference pictures”). This is consistent with the ’877 Patent, which refers to the buffer as “memory,” ’877 Patent, 5:26-39 (“the reconstructed reference image [] is stored in DPB memory”), and how the Applicant during prosecution compared the claimed “decoded picture buffer” to a prior-art “reference frame store,” Ex-1002, 237, which was defined in the prior art as “any type of memory or data storage device to store video blocks reconstructed from previously encoded blocks.” Ex-1017, 10:3-6.

298. Furthermore, Fimoff’s Figure 5 shows that the resampling/motion-compensation-interpolation filter (104) obtains the reference images from memory (106), without storing any resized reference images as part of its filtering process:

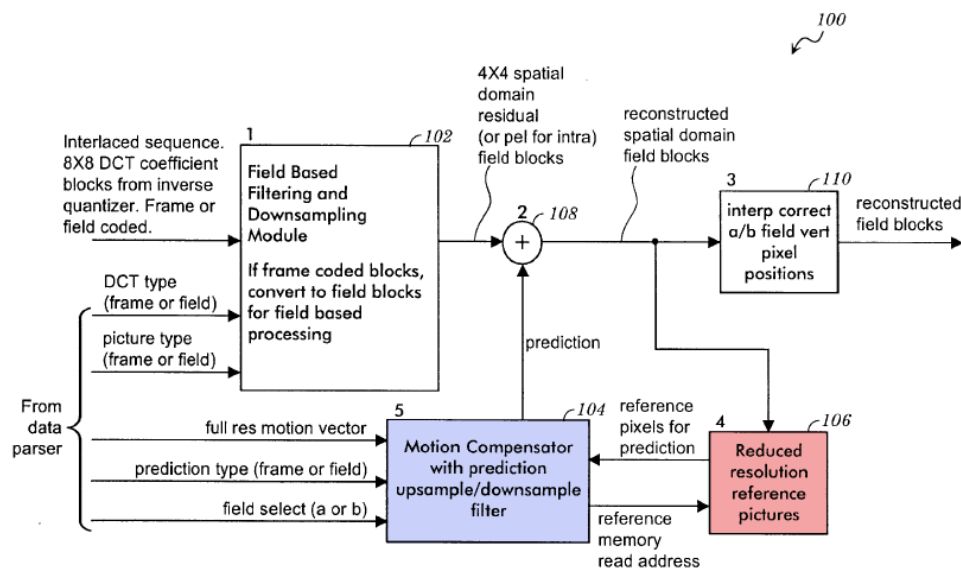


Figure 5

Specifically, although Fimoff’s decoder receives high-definition images from the encoder, the decoder does not reconstruct frames in high definition and does not

store them that way; all reconstructed reference images are stored in “reduced resolution.” *See, e.g.*, Fimoff, Fig. 5 (showing only “reduced resolution reference pictures” being stored). When the reference pictures are upsampled and the motion vectors are applied, those upsampled versions of the reference image are *not* stored in memory. *See id.* (showing a combined upsample/downsample filter). Instead, the combined resampling/motion-compensation-interpolation filter (104) outputs a standard-definition image with its associated motion vectors, which is combined with a (standard-definition) residual that is output by a separate downsampling module (102). It is that combined, fully reconstructed image (in standard-definition or “reduced resolution”) that is stored in memory as the next reference frame. Fimoff, 30:34-38. Fimoff never stores a resampled, different-sized version of the reference frame in memory.

299. In my opinion, a POSITA would have further recognized that there was no need to store a resampled version of the reference image because Fimoff proposed performing the resampling and motion compensation operations in a single operation. A POSITA therefore would have understood that, upon modifying Wiegand and Davies to incorporate Fimoff’s filtering process, the incorporation of Fimoff’s filters would have avoided any risk of overloading the decoded picture buffer associated with the resampling operation that Davies had originally proposed (and of which Misra had warned).

2. Claim 4

300. In my opinion, the combination of Wiegand, Davies, and Fimoff teaches or suggests all of the elements of claim 4.

301. In my opinion, a POSITA would have understood that Wiegand's draft HEVC standard is focuses on describing the operation of the decoder and pays less attention to describing the encoder. *See* Wiegand, 5-8 (dedicating a section (§8) for a "Decoding process," but no similar section for an encoder). But a POSITA would have understood that the draft standard's focus on the decoder was for a specific reason: coding standards provide the syntax for a bitstream to ensure that any encoded signal that complies with the standard could be decoded by any standard-compliant decoder. In other words, it is a way to ensure interoperability between standard-compliant devices. For this reason, specific details of how a particular encoder produces a standard-compliant bitstream are less important within the context of standardization than ensuring that any compliant decoder will be able to interpret and correctly decode the bitstream. For example, encoder vendors are free to develop their own individual algorithms for, e.g., how the encoder searches for a best matching block in a reference frame and how it decides which encoding mode to use for each individual coding unit or block of an input frame—so long as those choices are signaled and encoded into the bitstream using standard-compliant syntax. By contrast, the details of how a decoder interprets and decodes those syntax

elements must be standardized so that every compliant decoder decodes a compliant-encoded bitstream in the same way. Moreover, this implies that certain aspects of an encoder such as the “in-loop” decoding process for producing reference frames at the encoder must be specified by the standard to ensure that the reference frames decoded and stored in the reference frame buffer at the encoder exactly match those that would be obtained by any standard-compliant decoder.

302. Despite this more-liberal approach to encoders, at a high level, a POSITA would have understood that encoders perform most of the same processes as the decoder, but in reverse. *See, e.g.*, Fimoff, 2:47-50 (“The decoding process implemented by the MPEG-2 decoder 30 can be thought of as the reverse of the encoding process implemented by the MPEG-2 encoder 10.”). In other words, an encoder will take an image as its input, and apply a series of operations to compress and encode the image into a bitstream. When the decoder receives this bitstream, it will then reverse the encoder’s operations to reconstruct as close an approximation to the original image as possible given the quantization parameter choices that were made at the encoder.

303. While an encoder has access to both the original version of each video frame and the encoded version, the decoder cannot see the original version and has access only to the encoded version that it receives in the encoded bitstream. This is the reason that, as I just explained above, an encoder must implement an “in-loop”

version of the decoder (typically omitting the entropy decoder which is a lossless component) in order to replicate, at the encoder, the same decoded reference frames that will be obtained at the decoder. This is shown in Fimoff's Figures 1 and 2, which depict a generic MPEG-2 encoder and decoder, respectively:

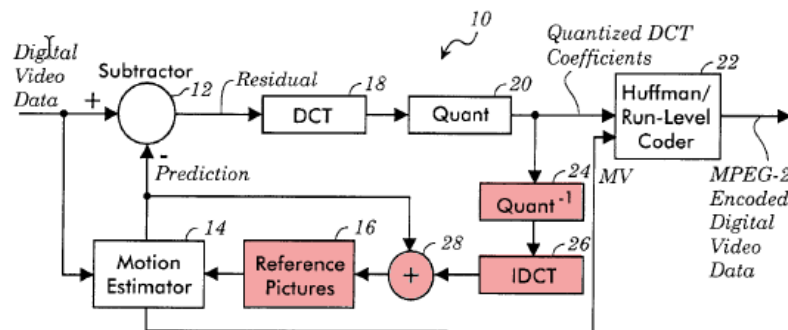


Figure 1

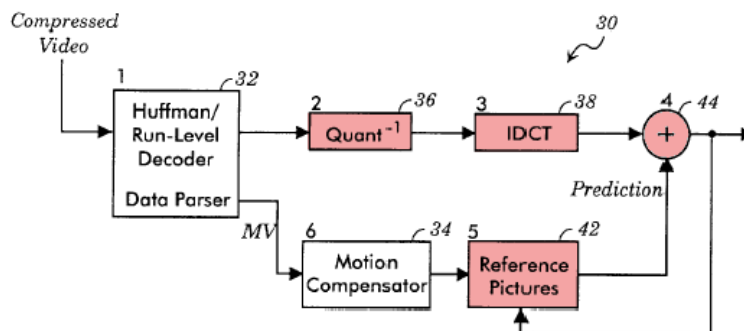


Figure 2

The modules shown in red appear in both the encoder and decoder, except the encoder implements these modules as part of a feedback loop. Fimoff, 2:20-23. As Fimoff explains, when a new macroblock enters the encoder, it is sent to both a subtractor (12) and motion estimator (14). *Id.*, 1:48-50. The motion estimator then “compares each new macroblock to be encoded with the macroblocks in a reference

picture previously stored in a reference picture memory 16.” *Id.*, 1:51-54, and selects a best-matching macroblock from the reference picture to use as an interframe prediction for the new macroblock to be encoded. The subtractor (12) then subtracts this prediction block from the new macroblock to obtain the *residual* (i.e., the prediction error between the selected prediction block and the new macroblock to be encoded), which is then processed and transmitted in the encoded bitstream. *Id.*, 1:50-64. The encoder must then use the red-shaded “in-loop” decoder modules shown in annotated Figure 1 above to decode the new macroblock that was just encoded “in order to serve as a macroblock of a reference picture for predicting subsequent pictures.” *Id.*, 2:20-37. As Fimoff explains, “[t]he object of this internal loop is to have the data in the reference picture memory 16 of the MPEG-2 encoder 10 match the data in reference picture memory of the MPEG-2 decoder.” *Id.*, 2:33-37.

304. In fact, those in the art had taken great pains to ensure the encoder and decoder produced as close-to-identical reference images as possible to maintain both accurate predictions and residuals based on those reference images. Otherwise, decoders risked “drifting” from the parameters the encoder was using for prediction and residual calculations, which would result in poor reconstruction results. *See, e.g., See, e.g., Ex-1033, 73* (“It is very important that both encoder and decoder use identical reference frames to create the prediction. Otherwise, the predictions in

encoder and decoder may be different, leading to an increased error between [the two].”); Ex-1032, 231 (“the input signals to the predictor at the encoder and the reduced resolution decoder can diverge, resulting in a loss of signal quality. In the MPEG framework, this behaviour is termed drift.”); Fimoff, 7:61-65 (building in “minimum drift prediction filtering” operations to avoid drift). For instance, if the reference image used by the encoder and decoder differed, neither the motion vectors nor residual calculated by the encoder may line up exactly right with the reference image in the decoder.

305. As I previously explained, a POSITA would have been motivated to produce and had a reasonable expectation of success in producing a *decoder* that would reconstruct a video from a bitstream with different-sized images (to account for network conditions), as taught by Wiegand, Davies, and Fimoff. *See generally* Section IX.B.1. Upon producing such a decoder, a POSITA would have also been motivated to produce an *encoder* to create a bitstream compatible with the decoder because, without a compatible encoder, there would be no bitstream to decode.

306. As part of the encoder, a POSITA would have understood that the encoder would need to replicate the decoding process as part of a decoder loop to ensure the reference frames used for predictions by the encoder and decoder match, for the reasons I just explained. Therefore, a POSITA would have understood that the encoder, like the decoder, would need to resample and perform motion

compensation on reference frames when predicting a different-sized current frame and calculating the residual. To accomplish this, a POSITA would have been motivated to implement the same combined resampling/motion-compensation interpolation filter operation used in the decoder taught by Wiegand, Davies, and Fimoff as part of the encoder's decoder loop to ensure the reference frames generated by both the encoder and decoder are the same. In summary, a POSITA would have known how to produce an encoder that was compatible with the decoder taught by Wiegand, Davies, and Fimoff.

307. In my opinion, a POSITA would have had a reasonable expectation of success in producing an encoder compatible with the decoder taught by Wiegand, Davies, and Fimoff because producing compatible encoders were routine and well within the skill of a POSITA. Indeed, Wiegand's silence as to the specific implementation of the encoder demonstrates that a POSITA could produce one knowing only the bitstream's syntax and the operation of the decoder. In this case, producing a compatible encoder would involve little more than implementing the decoder taught by Wiegand, Davies, and Fimoff in a decoding loop of a conventional predictive encoder. In doing so, a POSITA would be able to ensure the reference frames used by both the encoder and decoder match.

- a. **4[pre] – “A coding method of a current block of a current image from a reference block of a reference image reconstructed at a different size from the size of said current image, said reconstructed reference image being stored in a decoded picture buffer comprising:”**

308. In my opinion, the combination of Wiegand and Davies teaches or suggests element 4[pre]. *See* Sections IX.B.1.a (element 1[pre]), IX.B.2 (claim 4, generally).

309. Specifically, because Wiegand and Davies together teach a decoding method that reconstructs current image blocks based on blocks from different-sized reference images in memory, *see* Section IX.B.1.a (element 1[pre]), a POSITA would have motivated and reasonably expected to also provide an encoder that would perform a coding method for a current block based on a reference block of a different-sized reference image in order to create the bitstream for the decoder taught by Wiegand and Davies to operate on, *see* Section IX.B.2 (claim 4, generally). A POSITA would have understood, for instance, that because Wiegand and Davies’s combined decoder would decode current blocks based on different-sized reference images, so too would the encoder so that both the encoder and decoder would be using the same-sized reference images as part of their prediction and residual calculations. *See id.*; Fimoff, 2:33-37.

- b. 4[a] – “motion compensating said reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

310. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 4[a]. *See* Sections IX.B.1.b (element 1[a]), IX.B.2 (claim 4, generally).

311. As I explained previously, Wiegand and Davies collectively describe a decoder that receives different-sized frames sent from an encoder in response to changing network conditions. *See* Section IX.B.1.a (claim 1[pre]). Fimoff then provides a joint filter that performs both resampling and motion-compensation interpolation when decoding different-sized reference frames, which a POSITA would have recognized avoids the use of memory (or a buffer) to store resampled versions of the reference images. *See* Section IX.B.1.b (element 1[a]).

312. However, Fimoff is concerned with having legacy, standard-definition television sets support high-definition video streams. *See* Fimoff, 3:37-47. Thus, Fimoff provides only a decoding process that consistently receives and then converts high-definition images into standard-definition images; Fimoff has no control over the encoder generating the high-definition video images.

313. However, a POSITA would have appreciated that, when combined with Wiegand and Davies, Fimoff’s filters would also be employed at the encoder too.

Specifically, a POSITA would have recognized that the coding process taught by Wiegand and Davies would not be consistently transmitting high-definition content from an encoder to a decoder, like Fimoff. Instead, the decoding process taught by Wiegand and Davies would result in images of different sizes being sent from the encoder to the decoder in response to changing network conditions. *See* Section IX.B.1.b (element 1[a]). That means that the encoder must also encode images using different-sized reference frames, just like the decoder. But a POSITA would have also appreciated that, just like the decoder, the memory at the encoder is also size-constrained.

314. In my opinion, then, a POSITA would have been motivated to incorporate Fimoff's combined resampling/motion-compensation-interpolation filter into the encoder as well for two reasons: (1) to avoid overwhelming the limited buffer memory, just like the decoder; and (2) to ensure the (different-sized) reference pictures generated by the encoder match those as the decoder to provide accurate predictions and residual calculations for subsequent images (and avoid drift between the two). *See* Sections IX.B.1.b (element 1[a]), IX.B.2 (claim 4, generally).

c. 4[b] – “calculating, for the current block, a residue block from said current block and from said motion compensated reference block, and”

315. In my opinion, Wiegand discloses or suggests element 4[b].

316. Wiegand discloses that a “prediction residual” is calculated after accounting for the motion vectors as part of the *encoding* process. Wiegand, 14. In my opinion, a POSITA would have understood that the “prediction residual” is calculated by comparing the current block to the predicted, motion compensated reference block because that is how the residual error between the two is routinely calculated. *See, e.g.*, Fimoff, 1:48-2:6 (explaining how “motion estimator 14 reads this matching macroblock ... out of the reference picture memory 16 and sends it to the subtractor 12 which subtracts it ... from the new macroblock entering the MPEG-2 encoder 10. The output of the subtractor 12 is an error, or residual, that represents the difference between the predicted macroblock and the new macroblock being encoded.”). In fact, this was standard practice for calculating a residual block for encoding.

d. 4[c] – “coding the residue block in a binary stream,”

317. In my opinion, Wiegand discloses or suggests element 4[c].

318. Wiegand discloses that a “prediction residual” is “compressed using a transform ... before it is quantised,” which is then combined with the motion vectors “and encoded using either variable length coding or arithmetic coding” as part of the encoding process. Wiegand, 14. In my opinion, a POSITA would have understood that “variable length coding or arithmetic coding” refers to the encoding scheme of the residue block (and motion vectors) to convert it into a binary (bit)stream.

319. Specifically, a POSITA would have understood Wiegand's discussion of "variable length coding" and "arithmetic coding" to refer to general encoding processes to map pre-determined symbols to binary codewords. One simple example of variable-length coding is Morse code, which maps each of the letters of the alphabet to a set of binary dots or dashes. *See* Ex-1027, 1-3. In more-modern digital applications, these methods are employed to encode motion vectors and residuals with different combinations and lengths of binary values to minimize the amount of data needed to encode those features. *See, e.g.*, Fimoff, 1:37-42 (describing encoding motion vectors and residual using "Huffman" coding, which is a well-known type of variable-length entropy coding); *see also, e.g.*, Ex-1028 (providing an overview of Huffman coding); *see also* Ex-1027, 6-11 (describing variable-length encoding/decoding, generally).

320. Moreover, coding the residual block into a binary stream was also considered standard practice for encoding video signals. *See, e.g.*, Fimoff, 2:13-19 ("The [residual] data encoded by the coder 22 are combined with the motion vector data ... and the combined data are transmitted to a receiver that includes an MPEG-2 decoder 30."); *see also* Sita, ¶51 ("In the encoding process ... the blocks of transformed coefficients are further encoded by applying run-length and variable-length encoding techniques."); Ex-1014, ¶4 ("The video encoder applies transform,

quantization and entropy coding processes to further reduce the bit rate of the residual block.”).

- e. **4[d] – “wherein said single vertical filter GFv applied on a pixel s is such that $GFv(s)=MCIFv(SCFv(s))$, where MCIFv is a vertical motion compensation interpolation filter and SCFv is a vertical resampling filter, MCIFv and SCFv being applied jointly and”**

321. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 4[d]. *See* Sections IX.B.1.e (element 1[d]), IX.B.2 (claim 4, generally).

- f. **4[e] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”**

322. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 4[e]. *See* Sections IX.B.1.f (element 1[e]), IX.B.2 (claim 4, generally).

- g. **4[f] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”**

323. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 4[f]. *See* Sections IX.B.1.g (element 1[f]), IX.B.2 (claim 4, generally).

3. Claim 7

a. 7[pre] – “A decoding device comprising at least one circuit configured to:”

324. In my opinion, Wiegand discloses or suggests element 7[pre].

325. As a general matter, it was well-known in the art to implement decoders as software operating on a processor and memory.

326. Consistent with that general understanding, Wiegand discloses that the HEVC standard was intended to address a need “for higher compression of moving pictures for various applications such as videoconferencing, digital storage media, television broadcasting, internet streaming, and communication.” Wiegand, 13. Wiegand also explains that the HEVC standard would allow for “motion video to be manipulated as a form of computer data and to be stored on various storage media, [and] transmitted and received over existing and future networks.” *Id.*

327. Based on these descriptions, in my opinion, a POSITA would have understood that Wiegand’s decoding process was intended to operate on computers that contain at least a processor and memory. A POSITA would have further understood that the processor is a circuit made up of transistors and other components, and its coupling to memory forms an additional circuit because the data

signals passing from the processor to memory are conveyed through electrical circuitry.

- b. 7[a] – “access, from a decoded picture buffer, a reference image reconstructed at a size different from the size of a current image;”**

328. In my opinion, and for the reasons I discussed previously, the combination of Wiegand and Davies teaches or suggests element 7[a]. *See* Section IX.B.1.a (element 1[pre]).

- c. 7[b] – “motion compensate a reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

329. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 7[b]. *See* Section IX.B.1.b (element 1[a]).

- d. 7[c] – “decode, for a current block of said current image, a residue block from a binary stream, and”**

330. In my opinion, and for the reasons I discussed previously, Wiegand discloses or suggests element 7[c]. *See* Section IX.B.1.c (element 1[b]).

- e. **7[d] – “reconstruct the current block from said residue block and from said motion compensated reference block,”**

331. In my opinion, and for the reasons I discussed previously, Wiegand discloses or suggests element 7[d]. *See* Section IX.B.1.d (element 1[c]).

- f. **7[e] – “wherein said single vertical filter GF_v applied on a pixel *s* is such that $GF_v(s)=MCIF_v(SCF_v(s))$, where MCIF_v is a vertical motion compensation interpolation filter and SCF_v is a vertical resampling filter, MCIF_v and SCF_v being applied jointly and”**

332. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 7[e]. *See* Section IX.B.1.e (element 1[d]).

- g. **7[f] – “wherein said single horizontal filter GF_h applied on a pixel *u* is such that $GF_h(u)=MCIF_h(SCF_h(u))$, where MCIF_h is a horizontal motion compensation interpolation filter and SCF_h is a horizontal resampling filter, MCIF_h and SCF_h being applied jointly and”**

333. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 7[f]. *See* Section IX.B.1.f (element 1[e]).

- h. 7[g] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”**

334. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 7[g]. *See* Section IX.B.1.g (element 1[f]).

4. Claim 8

- a. 8[pre] – “A coding device comprising at least one circuit configured to:”**

335. In my opinion, and for the reasons I discussed previously, Wiegand discloses or suggests element 8[pre]. *See* Sections IX.B.2 (claim 4, generally) (encoder), IX.B.3.a (element 7[pre]) (circuit).

- b. 8[a] – “access, from a decoded picture buffer, a reference image reconstructed at a size different from the size of a current image;”**

336. In my opinion, and for the reasons I discussed previously, the combination of Wiegand and Davies teaches or suggests element 8[a]. *See* Section IX.B.1.a (element 1[pre]), IX.B.2 (claim 4, generally).

- c. **8[b] – “motion compensate a reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

337. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 8[b]. *See* Section IX.B.1.b (element 1[a]), IX.B.2 (claim 4, generally).

- d. **8[c] – “calculate, for a current block of said current image, a residue block from said current block and from said motion compensated reference block, and”**

338. In my opinion, and for the reasons I discussed previously, Wiegand discloses or suggests element 8[c]. *See* Section IX.B.2.c (element 4[b]).

- e. **8[d] – “code the residue block in a binary stream,”**

339. In my opinion, and for the reasons I discussed previously, Wiegand discloses or suggests element 8[d]. *See* Section IX.B.2.d (element 4[c]).

- f. **8[e] – “wherein said single vertical filter GFv applied on a pixel s is such that $GFv(s)=MCIFv(SCFv(s))$, where MCIFv is a vertical motion compensation interpolation filter and SCFv is a vertical resampling filter, MCIFv and SCFv being applied jointly and”**

340. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 8[e]. *See* Section IX.B.1.e (element 1[d]), IX.B.2 (claim 4, generally).

- g. 8[f] – “wherein said single horizontal filter GFH applied on a pixel u is such that $GFH(u)=MCIFH(SCFH(u))$, where MCIFH is a horizontal motion compensation interpolation filter and SCFH is a horizontal resampling filter, MCIFH and SCFH being applied jointly and”**

341. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 8[f]. *See* Section IX.B.1.f (element 1[e]), IX.B.2 (claim 4, generally).

- h. 8[g] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”**

342. In my opinion, and for the reasons I discussed previously, the combination of Wiegand, Davies, and Fimoff teaches or suggests element 8[g]. *See* Section IX.B.1.g (element 1[f]), IX.B.2 (claim 4, generally).

C. Ground 3: Obviousness over Wiegand in view of Davies and Sita

343. In Ground 2, I identified Fimoff as one example of a video-coding system that uses a joint resampling/motion-compensation-interpolation filter to predict current frames based on different-sized reference frames and I explained how that would have motivated a POSITA to modify Wiegand and Davies accordingly. *See, e.g.*, Section IX.B.1.b (Ground 2, element 1[a]). However, Fimoff is not the only reference that provides such a disclosure or motivation; as I mentioned previously, Sita also teaches a video-coding system that uses a joint resampling/motion-compensation-interpolation filter to predict current frames based

on different-sized reference frames. *See, e.g.*, Section IX.A.1.b (Ground 1, element 1[a]). In my opinion, these references have many similarities (and a few important differences). However, a POSITA could have used either one of them in combination with Wiegand and Davies.

344. Because the teachings of both Sita and Fimoff have already been described in detail, I will focus only on the elements in Ground 2 that rely on Fimoff, and explain why Sita would have led to the same result. For this reason, I will only address the claim elements related to motion compensation (elements 1[a], 4[a], 7[b], and 8[b]), the particular filters (elements 1[d]-[e], 4[d]-[e], 7[e]-[f], 8[e]-[f]), and the frame memory (elements 1[f], 4[f], 7[g], and 8[g]) because only those elements rely on Fimoff's teachings in Ground 2. And to the extent I cite Fimoff for other elements of Ground 2 to show how a POSITA would understand the state of the art, *see, e.g.*, Section IX.B.1.c (citing Fimoff to show the conventional nature of decoding a residual), in my opinion, Sita also provides substantially similar disclosures, as discussed with respect to each element in Ground 1.

1. **1[a], 4[a], 7[b], and 8[b] – “motion compensating said reference block of said reconstructed reference image by applying a single horizontal filter GFH and a single vertical filter GFv successively on the lines and on the columns of pixels of said reference block,”**

345. In my opinion, the combination of Wiegand, Davies, and Sita teaches these claim elements.

346. As I explained in Ground 1, in my opinion, a POSITA would have recognized that combining Wiegand and Davies would have resulted in a system that performs resampling and motion-compensation-interpolation operations, but that resampled versions of the reference images may need to be stored in the reference buffer that may already be at risk of overloading. *See* Section IX.B.1.b (Ground 2, element 1[a]). However, a POSITA would have known that Fimoff was not the only one to have solved this problem; Sita did as well.

347. As I explained previously, Sita teaches a decoder that receives encoded, high-definition video signals, and decodes those signals to produce a standard-definition video output. *See* Sita, ¶¶2-3, 66-67. As part of the decoding process, Sita downsamples and stores all reference frames in standard definition to reduce the burdens on the buffer, *id.*, ¶83, and then resamples the standard-definition reference frames back into high-definition, as needed, to apply the high-definition motion vectors received from the encoder as part of the motion-compensation process. *Id.*, ¶¶88-89. This is exactly what Fimoff teaches as well. However, compared to Fimoff, Sita is less concerned with the particularities of interlaced video, and Sita expressly recognizes that the resampling/motion-compensation-interpolation operations can be performed using separate horizontal and vertical filters. *Id.*, ¶95. In sum, for the reasons discussed previously, Sita teaches a motion-compensation process that uses different-sized reference frames by applying a horizontal filter and vertical filter to

the lines and columns of pixels in the reference block, as claimed. *See* Section IX.A.1.b (Ground 1, element 1[a]).

348. In my opinion, therefore, a POSITA would have been motivated to modify Wiegand and Davies's decoder to combine its resampling and motion-compensation-interpolation operations into a "single filter," as Sita does, to avoid overloading the buffer. Moreover, by combining the operations together, a POSITA would recognize that less computational resources would be needed because Sita expressly discloses that as one of the benefits of a combined operation. Sita, ¶110.

349. Furthermore, in my opinion, a POSITA would have had a reasonable expectation of success in making this modification because Sita already provided all of the equations to implement the filters as one, which would have required only

familiarity with computer programming within the ordinary skill level of those in the art.

2. 1[d]-[e], 4[d]-[e], 7[e]-[f], 8[e]-[f] – “wherein said single vertical filter GF_v applied on a pixel s is such that $GF_v(s)=MCIF_v(SCF_v(s))$ /, where $MCIF_v$ is a vertical motion compensation interpolation filter and SCF_v is a vertical resampling filter, $MCIF_v$ and SCF_v being applied jointly and wherein said single horizontal filter GF_h applied on a pixel u is such that $GF_h(u)=MCIF_h(SCF_h(u))$, where $MCIF_h$ is a horizontal motion compensation interpolation filter and SCF_h is a horizontal resampling filter, $MCIF_h$ and SCF_h being applied jointly and”

350. In my opinion, the combination of Wiegand, Davies, and Sita teaches these claim elements.

351. As I previously explained, a POSITA would have understood that Sita teaches the particular arrangement of the claimed vertical and horizontal filter (e.g., that each performs resampling and motion-compensation-interpolation filter operations jointly). *See* Sections IX.A.1.e and IX.A.1.f (Ground 1, elements 1[d] and 1[e]). And, as I also explained previously, in my opinion, a POSITA would have been motivated based on Sita’s teachings to combine the separate resampling and motion-compensation-interpolation operations taught by Wiegand and Davies to avoid needing to store resampled reference images in the buffer. *See* Section IX.C.1.

3. 1[f], 4[f], 7[g], and 8[g] – “wherein no resampled version of said reconstructed reference image is stored in the decoded picture buffer.”

352. In my opinion, the combination of Wiegand, Davies, and Sita teaches these claim elements.

353. As I previously explained, Sita teaches a joint resampling and motion-compensation-interpolation filter operation that avoids storing resampled versions of the reconstructed reference image in a buffer. *See* Section IX.A.1.g (Ground 1, element 1[f]). Accordingly, a POSITA would have recognized that, upon combining Sita with Wiegand and Davies, no resampled versions of a reference image would need to be stored in memory, either.

X. SUMMARY AND ADDITIONAL REMARKS

354. It is my opinion that Grounds 1, 2, and 3 each renders the Challenged Claims unpatentable as obvious, as I explained in the paragraphs preceding.

355. I currently hold the opinions expressed in this declaration based on the information available to me at this time. I may, however, form additional or new opinions should additional information or insights be made available, or upon further consideration.