UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SNAP INC. AND HISENSE USA CORPORATION,
Petitioners

v.

NOKIA TECHNOLOGIES OY,
Patent Owner.

IPR 2025-01375
U.S. Patent No. 11,805,267

PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 11,805,267

TABLE OF CONTENTS

EXHIBIT LIST

Exhibit No.	Description
1001	U.S. Patent No. 11,805,267 to Ugur et al. ("the '267 Patent")
1002	Prosecution File History for the '267 Patent
1003	Declaration of Immanuel Freedman
1004	U.S. Patent Application Publication No. 2010/0322303 ("Wada")
1005	U.S. Patent Application Publication No. 2011/0007799 ("Karczewicz-I")
1006	Yan Ye et al., High Precision Interpolation and Prediction, VCEG-AI33, July 2008 ("AI33 submission")
1007	U.S. Patent Application Publication No. 2009/0257499 ("Karczewicz-II")
1008	Sridhar Srinivasan et al., An Overview of VC-1, Proceedings of SPIE, Vol. 5960 (July 2005) ("Srinivasan")
1009	U.S. Patent No. 9,344,744 ("Kirchhoffer-744")
1010	U.S. Patent Application Publication No. 2003/0112864 ("Karczewicz-864")
1011	Wiegand, Overview of the H.264/AVC Video Coding Standard
1012	Richardson, The H.264 Advanced Video Compression Standard
1013	U.S. Patent Application Publication No. 2008/0198935 ("Srinivasan-935")
1014	ITU-T Recommendation H.264.2: Reference software for H.264 advanced video coding (March 2005)
1015	U.S. Patent Application Publication No. 2013/0034158 ("Kirchhoffer-158")
1016	U.S. Patent No. 8,594,188 ("Demos")
1017	IPR2024-00626, Paper 13: Decision Granting Institution of Inter Partes Review
1018	IPR2024-00627, Paper 16: Decision Granting Institution of Inter Partes Review
1019	U.S. Patent Application Publication No. 2008/0181308
1020	Paulo J. Cordeiro et al., Efficient Constrained Video Coding for Low Complexity Decoding (2008)
1021	Patent Owner's Index for Complaint Appendix C in ITC Inv. No. 337-TA-1448

1022	Patent Owner’s Appendix C0074 to Complaint in ITC Inv. No. 337-TA-1448
1023	Takeshi Chujoh et al., Internal Bit Depth Increase for Coding Efficiency, VCEG-AE13, January 2007
1024	Excerpts from Barry G. Haskell et al., Digital Video: An Introduction to MPEG-2 (Chapman and Hall, 1997)
1025	Joint Claim Construction Chart in ITC Inv. No. 337-TA-1448
1026	Declaration of Sylvia Hall-Ellis, Ph.D. (“Hall-Ellis”)
1027	Thomas Sikora, MPEG Digital Video-Coding Standards, IEEE Signal Processing Magazine (September 1997)
1028	Gary J. Sullivan and Thomas Wiegand, Rate-Distortion Optimization for Video Compression, IEEE Signal Processing Magazine (November 1998)

The ’267 patent takes a well-known technique—using higher precision intermediate values during interpolation—and applies it to a particular type of interpolation: biprediction. But Wada (Ground 1) already disclosed using higher precision pixel values to generate weighted predictions in connection with H.264. Additionally, the Karczewicz references (Ground 2) taught the averaging of higher-precision pixel values to generate more accurate subpixel predictions. Accordingly, the challenged claims would have been obvious to a POSITA and never should have issued.

The Board previously instituted a pair of IPR petitions based on the same Karczewicz references presented here in Ground 2. Ex. 1017, 1018. These IPR cases were terminated prior to final written decision due to settlement.

I. BACKGROUND

A. Inter Prediction and Bidirectional Interpolation

Well before the ’267 patent, video compression used “motion-compensated prediction” or “inter prediction” to exploit temporal redundancy between video frames. In unidirectional prediction, ‘P’ macroblocks in a picture were predicted using a block of pixels from a different “reference” picture:

MOTION-COMPENSATED PREDICTION

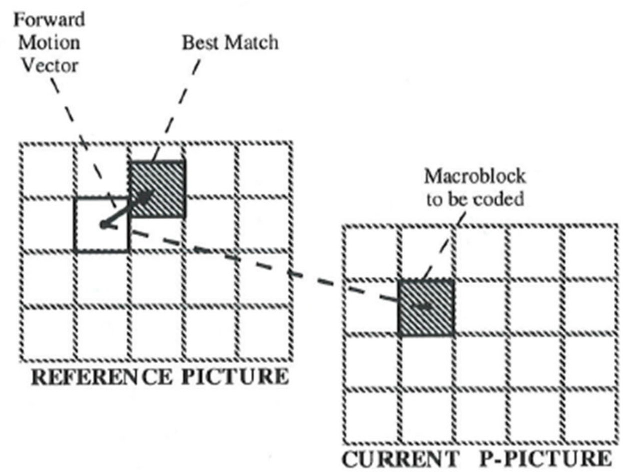


Fig. 7.1 Motion-Compensation using Unidirectional prediction. A displaced macroblock in a previous reference picture is used as the prediction of the moving-area macroblock in the current P-picture. The prediction error signal is then coded and transmitted.

Ex. 1024, 147. A motion vector describing the displacement of the reference block was coded. *Id.* Additionally, the difference between the current macroblock and the prediction ("prediction error" or "residual") was coded. The decoder used the motion vector to locate the reference block and added the residual to reconstruct the current macroblock. *Id.*; *see also* Ex. 1027, 85.

In bidirectional prediction, 'B' macroblocks in the current picture were predicted from reference blocks in two different reference pictures:

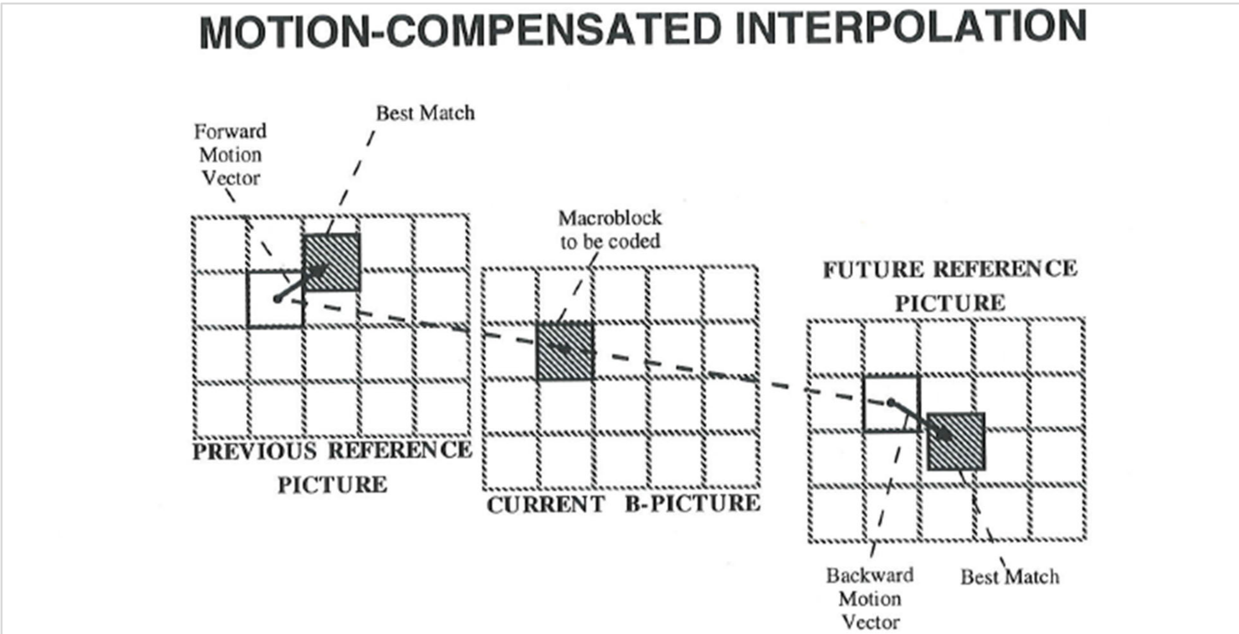


Fig. 7.2 Motion-Compensated Interpolation using Bidirectional prediction. A displaced macroblock in the previous picture is used as one prediction of the moving-area macroblock in the current B-picture. A displaced macroblock in the future picture is used as another prediction of the moving-area macroblock in the current B-picture. One, or the other, or an average of both can be used as the final prediction. The prediction error signal is then coded and transmitted.

Ex. 1024, 148. As the title in the figure above reflects, bidirectional prediction is a type of "interpolation" because the two reference blocks are used to *interpolate* the B macroblock. *Id.* ("Bidirectional temporal prediction, also called Motion-Compensated *Interpolation*, is a key feature of MPEG video."); Ex. 1027, 90 ("[T]he concept of B-pictures (bidirectional predicted/bi-directional *interpolated* pictures) was introduced in MPEG-1.").[1]

The H.264 standard (2003) supported different approaches for weighting reference blocks when interpolating the prediction. Ex. 1011, 000011; Ex. 1005, ¶¶43-46. The simplest approach, default weighted prediction, weighted the two

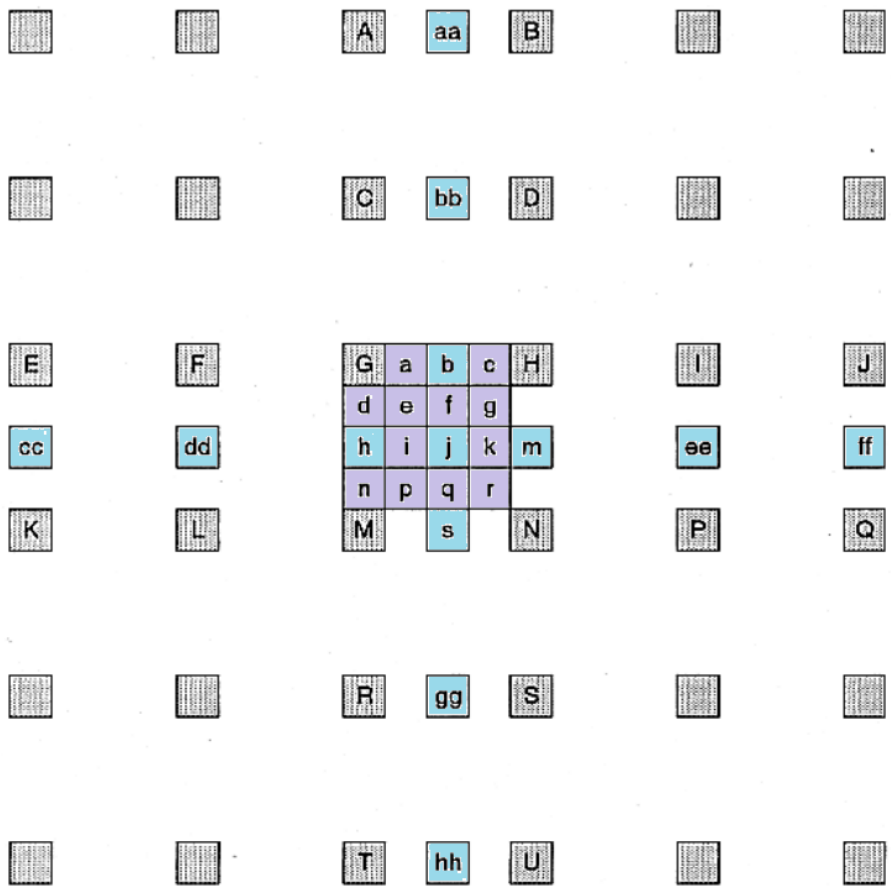
---

[1] All emphasis within quotations added by Petitioners unless otherwise noted.

reference blocks equally such that the interpolated prediction was simply the average of the pixels from the reference blocks. Ex. 1005, ¶¶44,60.

B. Subpixel Interpolation

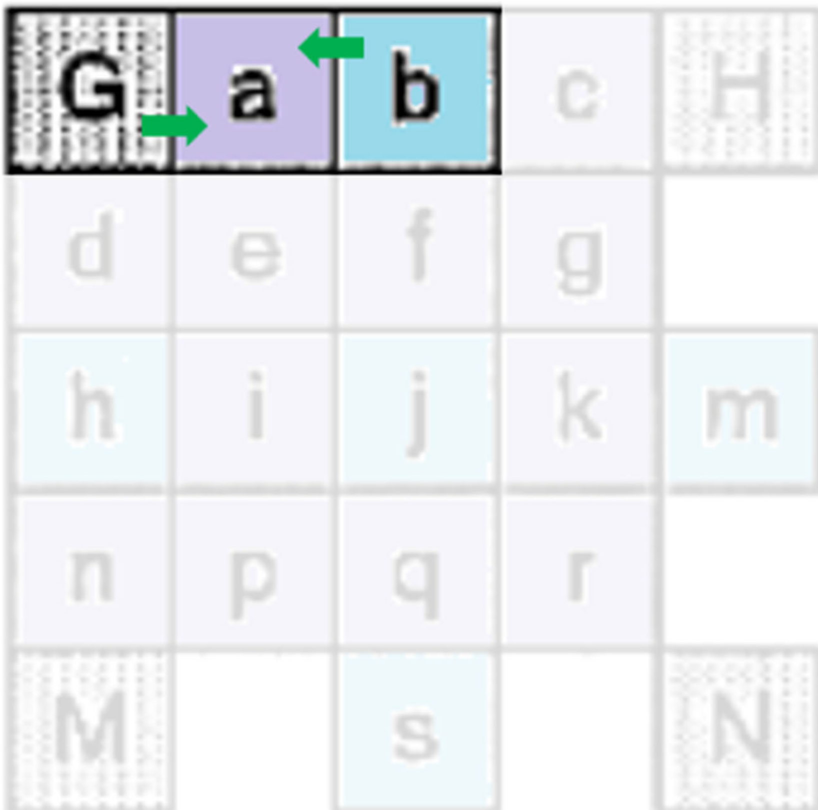
The best block in the reference image for predicting a block in the current frame does not always align with full pixel boundaries. As a result, video standards allowed motion vectors to point to "subpixels" located between the integer pixels (depicted below). In MPEG-1 (1993), motion vectors could point to half-pixel locations. Ex. 1028, 75 (discussing "half-pixel motion"). H.264 added support for motion vectors with quarter-pixel resolution. Ex. 1011, 000010. The figure below shows the relative locations of integer pixels (gray) as well as certain half-pixels (blue) and quarter-pixels (purple):



Ex. 1011, 000010 (Fig. 13).[2]

When a motion vector pointed to a subpixel, the subpixels making up the reference block had to be interpolated. Ex. 1028, 78 (half-pixel motion requires interpolation of half-pixels). In H.264, 6-tap filters were used to interpolate half-pixels from integer pixel values in the same row or column. Ex. 1011, 000010. The interpolated half-pixels were then used in turn to interpolate quarter-pixels. As an example, the quarter-pixel 'a' in the figure above was interpolated by averaging integer pixel G to its left and half-pixel b to its right. *Id.*

[2] All color annotations in figures added by Petitions for emphasis.



Id., Fig. 13.

C. Bit Shifting and Precision

As Dr. Freedman discusses, bit shifting has long been used in video compression as an efficient way to multiply and divide values by powers of 2. Ex. 1003, ¶15. However, when right-shifting is used for division, the remainder is discarded which can introduce rounding error. *Id.* Accordingly, calculations often included rounding adjustments. For example, the H.264 calculation for quarter-pixel 'a' discussed above adds a rounding adjustment of 1 before applying a right-shift of 1 bit to divide the sum by 2:

a = (G + b + 1) >> 1

Ex. 1011, 000010.

As noted, H.264 default weighted biprediction also averaged pixels from two reference blocks. The equation employs the same simple averaging approach used during quarter-pixel interpolation—two reference pixels are summed, a rounding adjustment of 1 is added, and the sum is right-shifted by 1, again, to divide by 2:

$$\text{pred}(i,j) = (\text{pred0}(i,j) + \text{pred1}(i,j) + 1) >> 1$$

Ex. 1005, ¶60; *id.*, ¶55 ("The addition of 1 is a rounding adjustment.").

Bit shifting was also used in video coding to adjust pixel/sample precision. For example, the 6-tap filters used in H.264 to interpolate half-pixels can generate values well beyond the [0 255] range for an 8-bit value. Accordingly, right-shifts of 5 bits were applied to reduce the precision of the final prediction values to 8 bits. Ex. 1011, 000010.

Before the alleged invention, it was well-understood that coding efficiency could be improved by using higher-precision intermediate values. *E.g.*, Ex. 1016, Abstract ("Image quality from MPEG-style video coding may be improved by preserving a higher number of bits during intermediate encoding and decoding processing steps."); Ex. 1009, 7:4-19 ("[B]y predicting and reconstructing in a higher precision than the picture is defined, a more precise prediction and reconstruction can be obtained[.]"), 17:11-24 (use of higher-precision intermediate values can improve predictions and reduce the size of residual); Ex. 1023, 1 ("In the field of digital signal processing, it is known that better results are generally able to be achieved by increasing the internal arithmetic precision to a higher level …."); Ex. 1008, 000015 ("It is possible to achieve additional accuracy by retaining a higher precision result after the first stage of filtering."); Ex. 1010, ¶62 ("maximum precision is retained" during quarter-pixel interpolation by "using intermediate values which are neither rounded nor clipped").

Notably, the H.264 standard itself used higher-precision intermediate values in certain interpolation calculations. For example, interpolating the half-pixel at the 'j' location involved applying a 6-tap filter to other interpolated half-pixels (as opposed to integer pixels). Rather than use the final, 8-bit prediction values for those half-pixels (*e.g.*, h), the standard calculated j1 using the higher-precision intermediate values (*e.g.*, h_1). Ex. 1011, 000010; Ex. 1007, ¶95. Ex. 1003, ¶20.

II. THE '267 PATENT

A. Overview

The '267 patent is directed to "utilizing motion prediction in video coding." Ex. 1001, Abstract. The patent discusses an "existing" process for generating a bi-directional prediction for a current block., Ex. 1001, 13:43-55, Fig. 10:

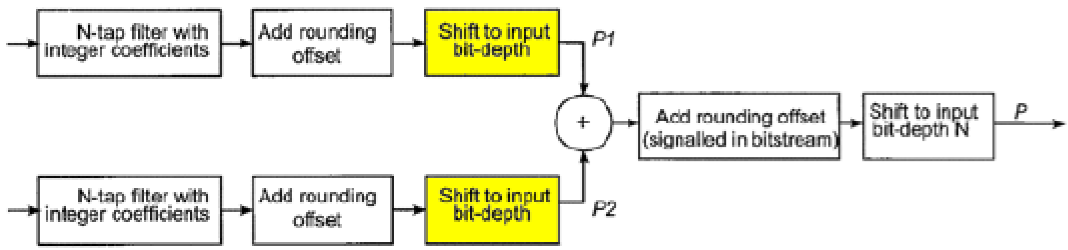


Fig. 10

The ’267 patent does not purport to invent this conventional process, admitting that the “Background” art includes video coding standards, *e.g.*, H.264 (Ex. 1001, 1:34-46), and “bi-directional prediction” (*id*., 3:49-55), with reference blocks determined by motion vectors (*id*., 2:20-34, 3:12-18), predictions determined based on reference blocks (*id*., 1:34-46), bi-directional prediction obtained by combining, two predictions (*id*., 3:49-55, 3:66-4:20), and residual data calculated as a difference between the prediction and current block (*id*., 1:52-59, 3:25-30, 2:1-12).

The patent further admits that the prior art included various techniques for addressing rounding errors when averaging two blocks. Ex. 1001, 3:66-4:20. According to the ’267 patent, “these methods increase somewhat the complexity” of encoders. *Id*., 4:21-25.

The patent purports to “reduc[e] the effect of rounding errors in bi-directional ... prediction” by maintaining prediction signals “in a higher precision during the prediction calculation” and rounding the prediction down to a lower precision "after the two or more prediction signals have been combined with each other." Ex. 1001, 4:29-43. In particular, it describes pixels having a first precision (e.g., N) and predictions having a second precision (*e.g.*, M) that is higher than the first. Ex. 1001, 12:41-13:55, Fig. 11:

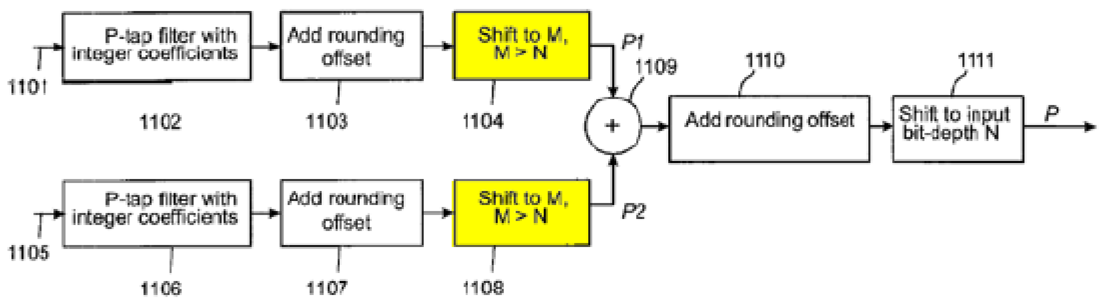


Fig. 11

The patent only purports to change the number of right-shifted bits for predictions before they are combined.

B. Priority Date

The '267 patent cites a provisional application filed January 7, 2011. This Petition analyzes obviousness as of that date, and all relied-upon art predates January 7, 2011.

C. Standing

Petitioners certify that they are not barred or estopped from requesting this *inter partes* review, and the '267 patent is eligible for it.

III. LEVEL OF ORDINARY SKILL IN THE ART

The field of the patent is video encoding/decoding. A POSITA at the time of the alleged invention of the '267 patent would have had a (1) Bachelor's degree in electrical engineering, computer engineering, computer science, or a comparable field of study such as physics, and (2) approximately two to three years of practical experience with video encoding/decoding. Additional experience can substitute for the level of education, and vice-versa. Ex. 1003, ¶¶29-31.

IV. CLAIM CONSTRUCTION

A. "prediction"

In IPR2024-00626/627, the Board partially construed the term "prediction" as "encompass[ing] values used for prediction that are calculated by mathematical operations including multiplying pixel values in reference blocks with weights." Ex. 1017, 16-17. The term was construed because Patent Owner argued that certain values the prior petitioner cited in the Walker prior art were "intermediate values" as opposed to "predictions." *Id.*, 12-13. Petitioners are neither relying on the Walker prior art nor on prediction values obtained by multiplying pixel values with weights. Accordingly, construing "prediction" is not necessary in this case.

B. "precision"

In IPR2024-00626/627, the Board noted in its institution decision there was "no dispute … that … precision can be the number of bits needed to represent possible values." Ex. 1017, 20. Petitioners agree that, in the context of the '267 patent, the term precision covers the number of bits needed to represent possible values. Ex. 1003, ¶34. For example, the '267 patent specification describes right-shifting values "so that the precision … becomes M bits." Ex. 1001, 13:10-12;

13:40-42. The claims provide further support. Claim 1 recites "decreasing [the] precision" of the combined prediction by right-shifting the bits. Dependent claim 4 recites using a left shift to get the second prediction with the claimed second (*i.e.*, higher) precision.

In the ITC case, Nokia is proposing a construction focused on "the number of distinct alternatives from which a value can be determined" to set up an argument that multiplying pixel values by a fixed weight (as in Walker) does not increase "precision" even if the weighted value requires more bits. Ex. 1025, 3. If Nokia proposes a similar construction here, the Board should reject it as unsupported.

V. IDENTIFICATION OF CHALLENGE

Petitioner requests *inter partes* review and cancelation of the challenged claims on the following grounds:

Grounds	Claims	Statutory Basis	Prior Art
1	1-36	§ 103	Wada in view of Karczewicz-I
1A	3, 9, 15, 21, 27, 33	§ 103	Wada in view of Karczewicz-I and Srinivasan
2	1-36	§ 103	Karczewicz-I in view of Karczewicz-II
2A	3, 9, 15, 21, 27, 33	§ 103	Wada in view of Karczewicz-I and Srinivasan

A. Grounds 1 & 1A

1. Wada (Ex. 1004)

Wada is U.S. Patent Application Publication 2010/0322303, filed August 10, 2010; it is prior art under §§102(a) and (e). Ex. 1003, ¶41.

Wada discloses coding techniques that optionally apply a restoration filter to improve the quality of reference image data used for prediction. Ex. 1004, Abstract; ¶¶17-19, 91-98, 127. Wada's "Seventh Embodiment" is most relevant here. *Id.*, ¶¶256-272. Figure 46 (below) shows encoding apparatus 7000 which includes prediction image generator 804.

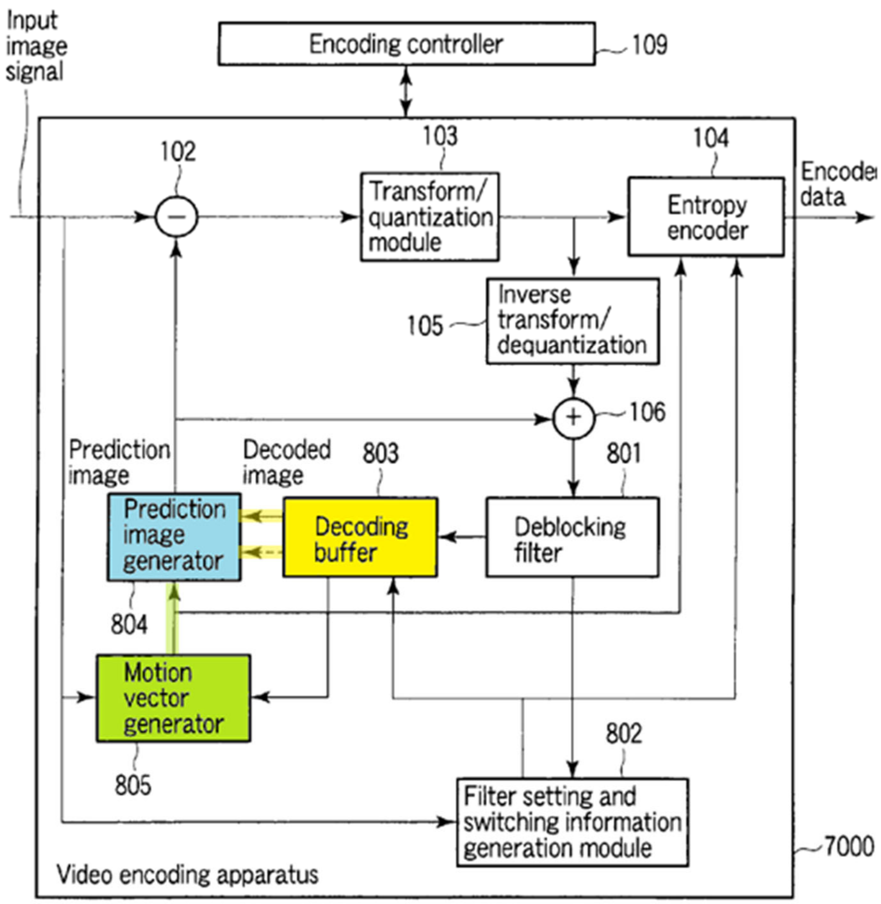
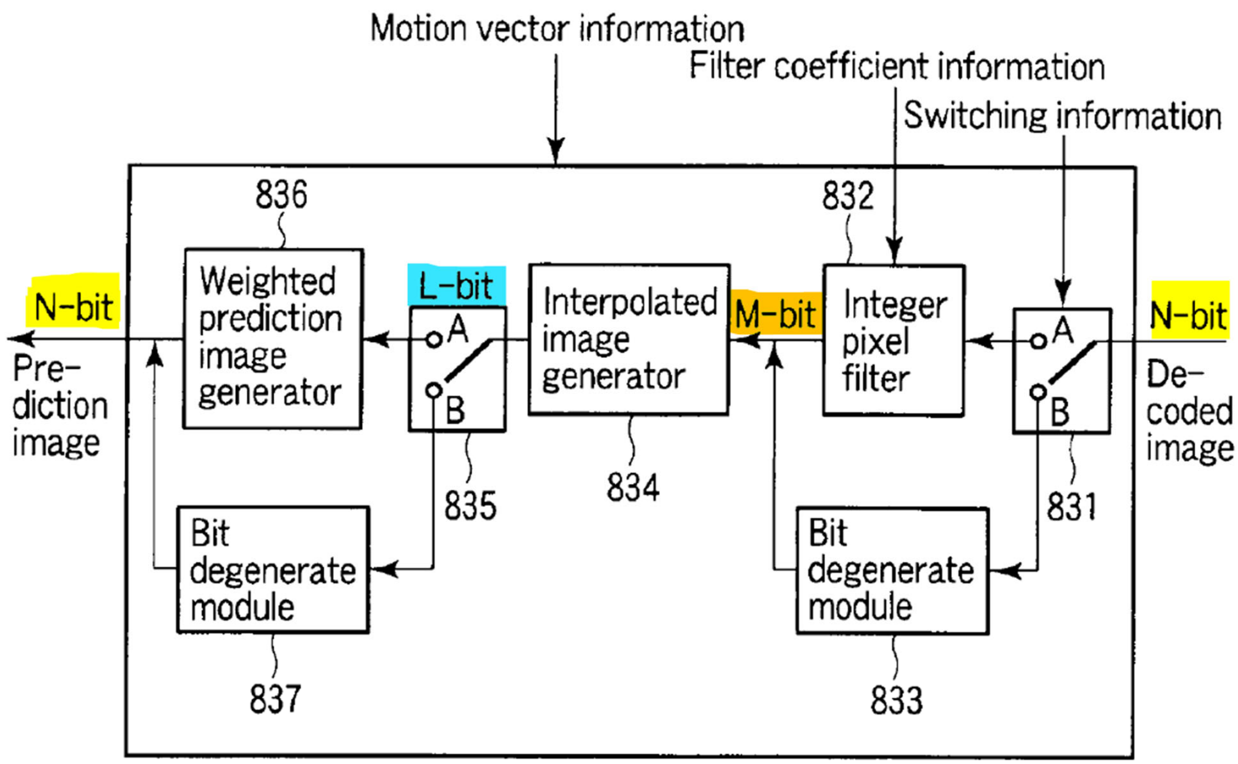


FIG. 46

Id., Fig. 46. "[P]rediction image generator 804 uses … motion vector information generated with the motion vector generator 805, and generates the motion-compensated prediction image." *Id.*, ¶262. Figure 47 shows a corresponding ***decoding*** apparatus 8000 with corresponding prediction image generator 814. *Id.*, ¶266, Fig. 47.

Figure 49 shows a "concrete implementation of the prediction image generators" from Figures 46 and 47. *Id.*, ¶269.



F I G. 49

Id., Fig. 49. As shown, the pixel bit length of the decoded image input (at the right) is N bits but intermediate image data has increased precision of M- and L-bits before precision is ultimately decreased back to N bits for the outputted prediction image (at the left). The increase from N to M bits can be performed by bit extension module 833 which applies a left shift. Ex. 1004, ¶270. "[T]he weighted prediction image generator 836 generates a prediction image based on a weighted prediction equation given in … AVC …." *Id.*, ¶271. AVC refers to Advanced Video Coding (*i.e.*, the H.264 standard). Ex. 1003, ¶43.

Wada is analogous art in the same field as the ’267 patent (*supra* §III) because it is directed to video encoding/decoding, motion prediction, and managing precision during inter prediction. Ex. 1003, ¶44. Like the ’267 patent, Wada was assigned the H04N subclass. Ex. 1004, cover.

2. Karczewicz-I (Exs. 1005, 1006)

Karczewicz-I is U.S. Patent Application Publication 2011/0007799, filed July 9, 2009; it is prior art under §102(e). Ex. 1003, ¶46.

Karczewicz-I teaches block-based techniques for motion prediction, including bi-directional predictions calculated by averaging predictions based on two reference blocks, which may be calculated using sub-pixel interpolation. *E.g.*, Ex. 1005, ¶35, ¶¶41-44, ¶¶55-60; Ex. 1003, ¶47.

Karczewicz-I incorporates by reference “a submission numbered VCEG-AI33, titled ‘High Precision Interpolation and Prediction,’ by Yan Ye, Peisong Chen, and Marta Karczewicz and presented at the 35th Meeting in Berlin, Germany, occurring on Jul. 16-18, 2008.” Ex. 1005, ¶37. Ex. 1006 is a copy of the AI33 submission that was filed with the Karczewicz-I application on July 9, 2009. Ex. 1006 (“AI33 submission”). Patent Owner filed the same document as Ex. 2010 in IPR2024-00626/627 to try to argue that Karczewicz-II was cumulative of prior art of record during prosecution. Patent Owner attached the same document to its latest ITC complaint because the AI33 submission was referenced in the prosecution history. Ex. 1021 (Index for Complaint Appendix C), 1022 (Appendix C0074). Accordingly, Ex. 1006 is the document Karczewicz-I incorporates by reference (and which Patent Owner itself disclosed during prosecution).

Karczewicz-I is analogous art in the same field as the ’267 patent (*supra* §III) because it is directed to video encoding/decoding, motion prediction, and H.264. *E.g.*, Ex. 1005, ¶89, ¶¶35-37; Ex. 1003, ¶49. Like the ’267 patent, Karczewicz-I was assigned the H04N subclass. Ex. 1005, cover.

3. Motivation to Combine

A POSITA would have been motivated to combine the teachings of Wada and Karczewicz-I for several reasons. First, as discussed, Wada discloses a weighted prediction image generator which applies a weighted prediction equation from the AVC standard to generate a weighted prediction image. Karczewicz-I describes weighted prediction equations for the AVC/H.264 standard. Ex. 1005, ¶¶57, 60, 61, 62, 64, 69. Second, beyond the equations, Karczewicz-I describes the different weighted prediction modes in H.264/AVC and how to select between them. *Id.*, ¶71, 95. These explanations would be relevant and helpful to a POSITA considering Wada’s prediction image generator. Ex. 1003, ¶50.

Third, Karczewicz-I describes techniques for improving on conventional H.264/AVC weighted prediction such as eliminating rounding adjustments where they degrade prediction accuracy. Ex. 1005, ¶¶22, 26, 47. Karczewicz-I also discloses techniques for improving weighted prediction by enabling the use of offsets even where the weight factors are based on default or implicit weighted prediction rather than explicit weighted prediction. *Id.*, ¶¶8, 28, 49, 69. Based on Karczewicz-I's teachings, a POSITA would have understood these weighted prediction improvements would improve the quality of the weighted prediction images generated by Wada's prediction image generator. Ex. 1003, ¶51.

A POSITA would have had a reasonable expectation of success in combining the teachings of Wada and Karczewicz-I as discussed. Wada expressly suggests using weighted prediction equations from the AVC standard and Karczewicz-I describes such equations. The use of Karczewicz-I's equations is consistent with Wada's teachings and would not have changed the principle of operation of either reference. Ex. 1003, ¶53.

The motivation for additionally combining Srinivasan for Ground 1A is discussed below in addressing claims 3, 9, and 15. *Infra* §V.A.4.c.

4. Encoder Claims

a. Independent Claims 1, 7, and 13

[1a]. A method for encoding a block of pixels, the method comprising:

Ground 1 teaches 1[a]. Ex. 1003, ¶¶55-57.

Wada teaches a method for encoding blocks of pixels, for example, in connection with encoding apparatus 7000 in Fig. 46. Ex. 1004, ¶¶256-262. Wada's encoding method uses a prediction image generator to generate motion-compensated prediction images. *Id.*, ¶262. Wada describes encoding and decoding macroblocks which are "block[s] of pixels." For example, Wada discloses a syntax structure including "a macroblock level syntax (1907) [that] recites transform coefficient data, prediction mode information and a motion vector which are needed for every macroblock." *Id.*, ¶102, Fig. 31.

Similar to Wada, Karczewicz-I describes encoding methods that use bidirectional prediction to encode blocks within video frames. Ex. 1004, ¶8 ("In bi-directional prediction, a video *block* may be predictively encoded and decoded based on two different lists of predictive reference data."); ¶52 ("As shown in FIG. 2, video encoder 50 receives a current video *block* within a video frame or slice *to be encoded*.").

[7a]. An apparatus for encoding a block of pixels, the apparatus comprising: at least one processor and at least one memory including computer program code, the at least one memory and computer program code configured to, with the at least one processor, cause the apparatus to:

Ground 1 teaches 7[a]. Ex. 1003, ¶¶58-59.

Wada discloses video encoding apparatus 1000 "can be realized using general-purpose computer equipment." Ex. 1004, ¶132. Wada further explains that the various encoder blocks "can be realized by making a processor mounted on the computer equipment execute a program" after, for example, "installing the program in the computer equipment beforehand." *Id.* Similarly, a POSITA would have understood from Wada that video encoding apparatus 7000 of Fig. 46 could likewise be implemented with conventional computer technology. Ex. 1003, ¶59. By the time of the alleged invention, a POSITA would have been well-aware that computers with processors, memory, and program code were routinely used to encode video. *Id.* By 2005, there was reference encoder and decoder software available for the H.264 standard. Ex. 1014. Similar to Wada, Karczewicz-I teaches that the described coding techniques can be implemented using processors and software stored in memory. Ex. 1004, ¶¶98-101.

[13a]. A computer program product for encoding a block of pixels, the computer program product comprising at least one non-transitory computer readable storage medium having computer executable program code portions stored therein, the computer executable program code portions comprising program code instructions configured to:

Ground 1 teaches 13[a]. Ex. 1003, ¶¶60-61.

As discussed for [7a], both Wada and Karczewicz-I teach computer-based encoders that execute installed software. Karczewicz-I specifically describes "a computer-readable medium comprising instructions that, when executed in a processor, performs" the described encoding methods. Ex. 1005, ¶99. Ex. 1003, ¶61.

[1b]/[7b]/[13b] [determining/determine], for a current block, a first reference

block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

Ground 1 teaches [1b], [7b], and [13b]. Ex. 1003, ¶¶62-66.

Wada discloses encoder 7000 with decoded image buffer 803 for storing local decoded images and motion vector generator 805. EX. 1004, ¶¶261-262. Prediction image generator 804 uses the local decoded images and the motion vector information from motion vector generator 805 to generate motion-compensated prediction images. *Id.*, ¶262.

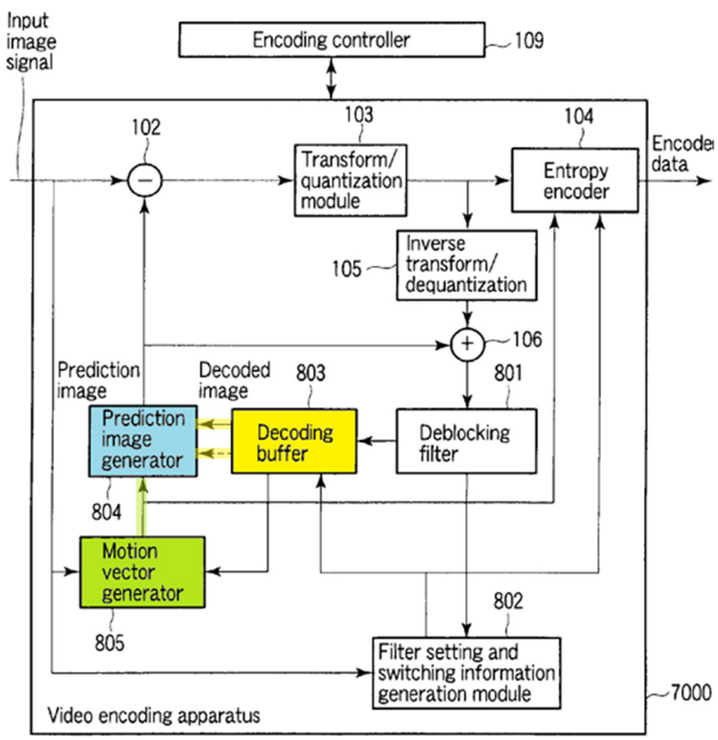


FIG. 46

Id., Fig. 46.

Wada's decoded image samples are N bits. Ex. 1004, ¶270.

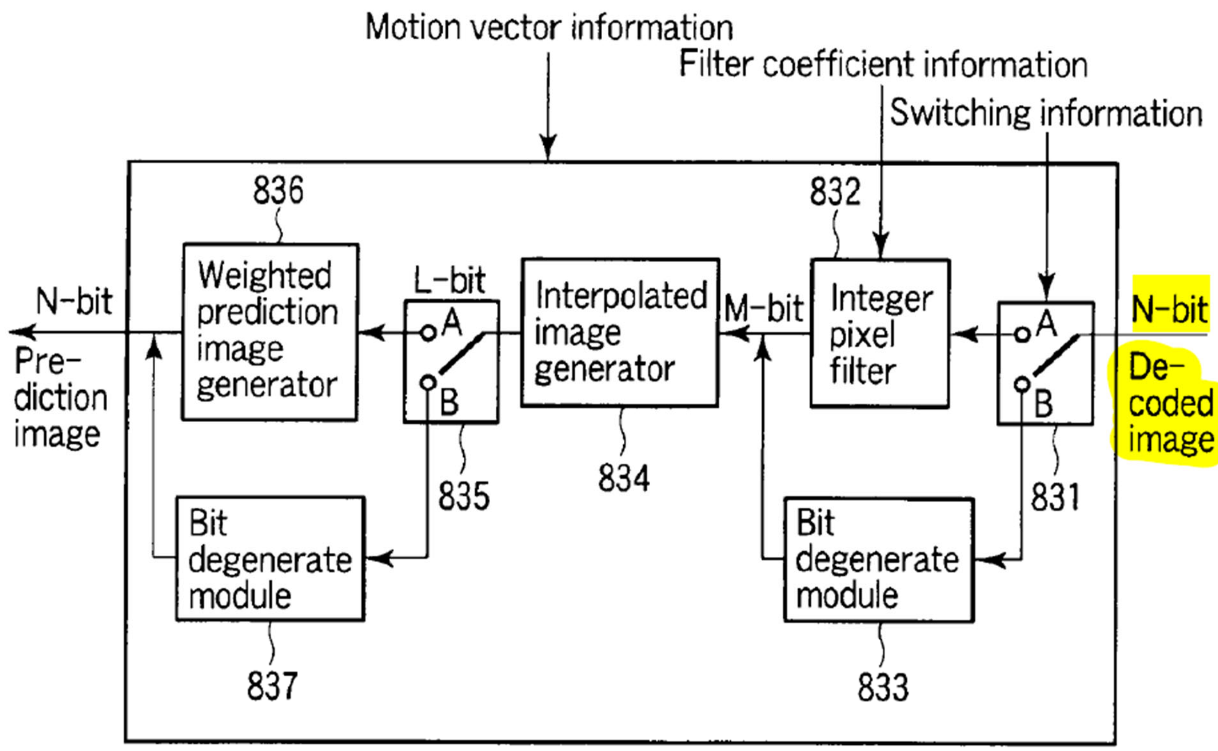


FIG. 49

Id., Fig. 49. This pixel bit length of N bits is a "first precision." Ex. 1003 ¶64.

Although Wada does not discuss biprediction in detail,[3] Karczewicz-I does. Karczewicz-I explains that "blocks in a B picture may be predicted" using "motion compensated prediction from the combination of [two] reference pictures" from "list 0" and "list 1." Ex. 1005, ¶58. Karczewicz-I also explains that motion compensation involves "generating the *predictive block based on the motion vector*

[3] As discussed, biprediction was well-known in the art (*supra* §I.A) and Wada does reference B pictures (Ex. 1004, ¶235) which a POSITA would have known use biprediction. Ex. 1003, ¶7.

determined by motion estimation." *Id.*, ¶53. A POSITA would have known that in bidirectional motion compensated prediction the blocks of reference pixels used to generate the combined prediction would either be integer pixels from the reference picture or subpixels interpolated based on integer pixels in the reference picture. *Supra* §I.B; Ex. 1011 at 000010; Ex. 1003, ¶14.

Accordingly, Wada and Karczewicz-I together teach or suggest determining first and second reference blocks based on motion vectors for use in generating a combined prediction of a current block and that the reference blocks have a first precision of N bits.

[1c]/[7c]/[13c] [using/use] said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[1d]/[7d]/[13d] [using/use] said second reference block to obtain a second prediction, said second prediction having the second precision;

Ground 1 teaches [1c], [7c], and [13c] as well as [1d], [7d], and [13d]. Ex. 1003, ¶¶67-71.

Wada discloses increasing the pixel bit length of the samples from the decoded image that are used to generate the prediction image. Ex. 1004, ¶270. The pixel bit length is M bits and L bits after the first and second stages of processing, respectively.

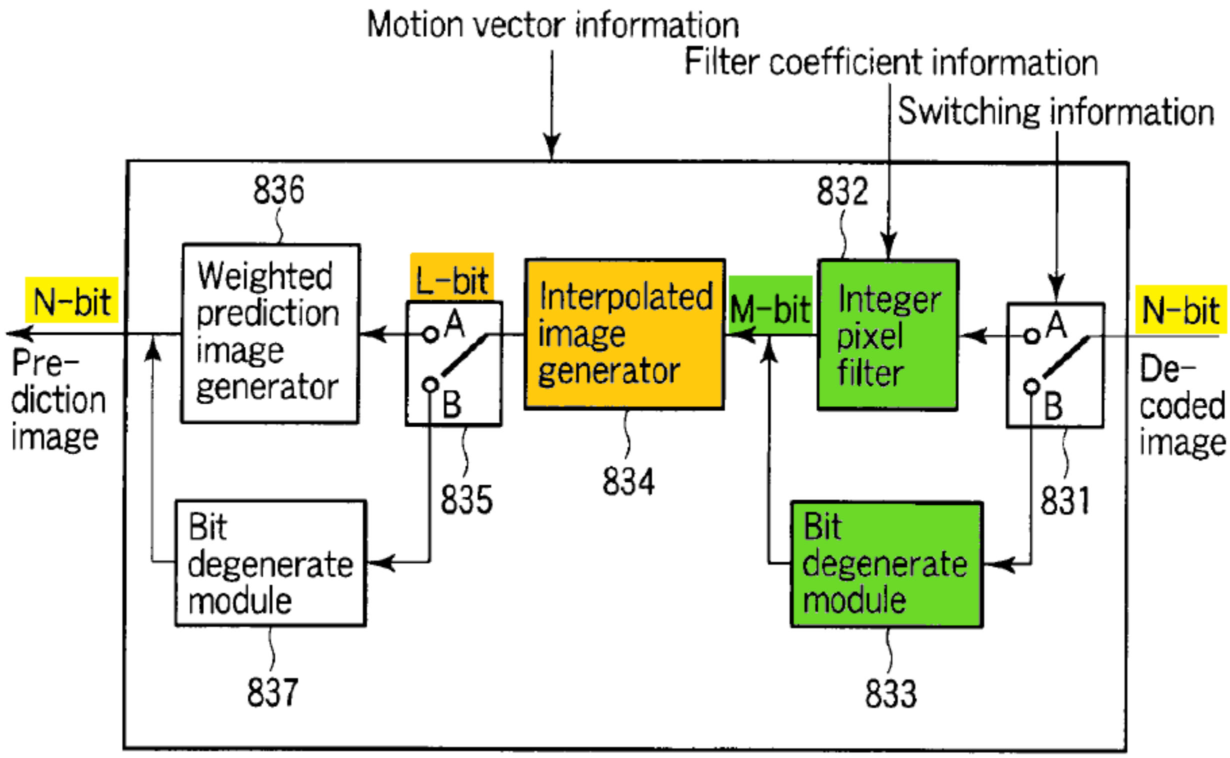


FIG. 49

Id., Fig. 49. In the first stage (green blocks), the pixel bit length is increased from N bits to M bits using either bit extension module 833[4] or integer pixel filter 832. Ex. 1004, ¶270. The bit extension module uses a left shift to expand the bit length of the pixel values from the decoded image from N to M bits. *Id.* Regardless of whether integer pixel filter 832 or bit extension module 833 is used, N-bit blocks

[4] Bit extension module 833 is mislabeled as "Bit degenerate module" in the figure. Ex. 1003, ¶68.

from a reference picture are used to obtain M-bit values having increased precision. Ex. 1003, ¶68.

In the second stage (orange block), interpolated image generator 834 generates L-bit prediction data. Ex. 1004, ¶271; Fig. 49. A POSITA would have understood that the decoded image data is used to obtain this higher-precision interpolated image data. Ex. 1003, ¶69. Relatedly, Karczewicz-I describes specific processes for interpolating higher-precision subpixels from integer pixels of a reference image. Ex. 1006.

The claimed "first prediction" and "second prediction" could be either the M-bit prediction data (green) output by the first stage of prediction image generation or the L-bit prediction data (orange) output by the second stage of prediction image generation. Ex. 1003, ¶70.

As discussed for [1b]/[7b]/[13b], Karczewicz-I discloses predicting B blocks using pixels from two reference pictures. Thus, it would have been obvious for Wada's prediction image generator to generate higher-precision predictions based on *two reference blocks* as claimed. Ex. 1003, ¶71.

[1e]/[7e]/[13e] [obtaining/obtain] a combined prediction based at least partly upon said first prediction and said second prediction;

Ground 1 teaches [1e], [7e], and [13e]. Ex. 1003, ¶¶72-76.

Wada's prediction image generator includes weighted prediction image generator 836 that "generates a prediction image based on a weighted prediction equation given in H.262/AVC, etc." Ex. 1004, ¶271.

Karczewicz-I discloses AVC/H.264 weighted prediction equations for predicting B video blocks. For example, Karczewicz-I discloses the following equation for bidirectional default weighted prediction:

$$\text{pred}(i,j) = (\text{pred0}(i,j) + \text{pred1}(i,j) + 1) >> 1$$

Ex. 1005, ¶60. Karczewicz-I notes that in the above formula "pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1." *Id.* Accordingly, pred(i,j) is a combined prediction based on two predictions—pred0(i,j) and pred1(i,j). Ex. 1003, ¶74.

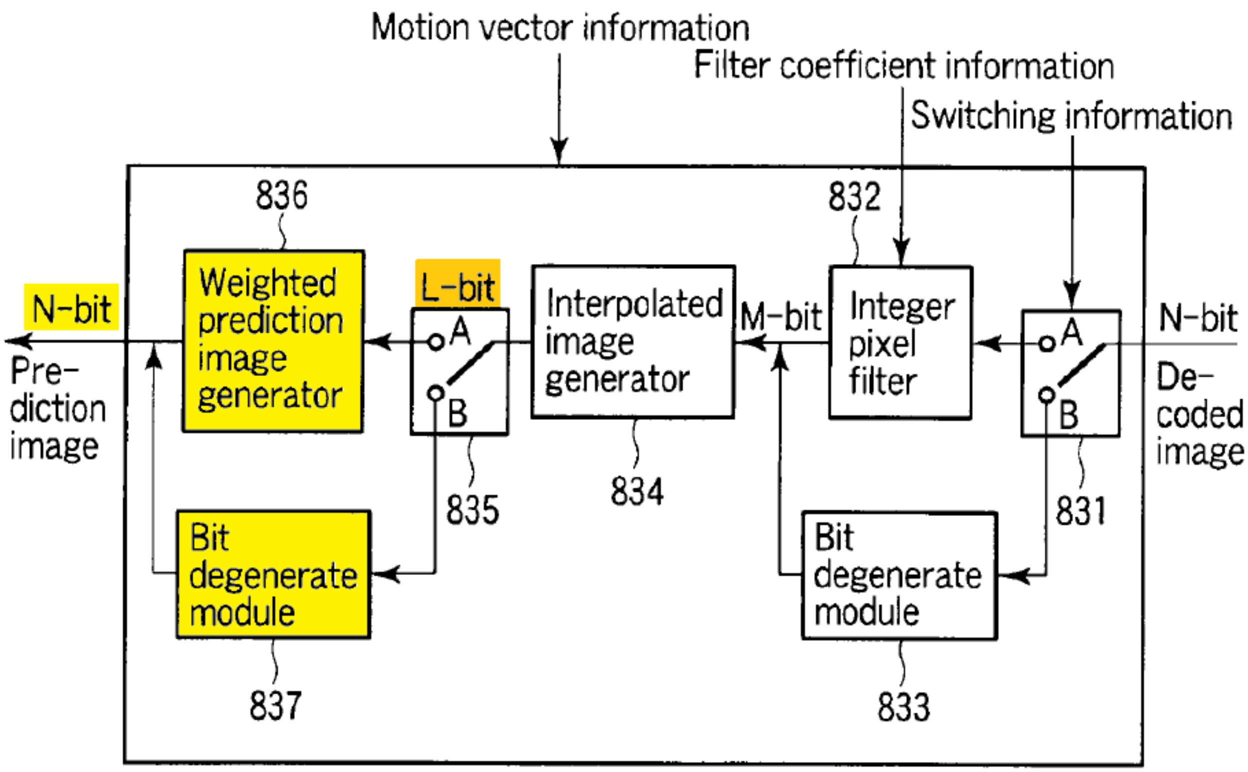
As discussed, it would have been obvious for Wada's prediction image generator to predict B video blocks by combining predictions obtained using reference blocks from two reference pictures consistent with Karczewicz-I's AVC-based weighted prediction equations and how biprediction was conventionally performed prior to the alleged invention. *Supra* §V.A.3; Ex. 1003, ¶75.

Moreover, as discussed for the prior limitations, the claimed "first prediction" and "second prediction" could be the L-bit prediction data output from Wada's interpolated image generator 834. That data feeds into weighted prediction image generator 836. Ex. 1004, Fig. 49. The claimed predictions could alternatively be the M-bit output of integer pixel filter 832 and/or bit extension module 833. The L-bit precision data that flows into weighted prediction generator 836 is based on that M-bit precision data output from the first stage. Accordingly, it would have been obvious that the combined prediction would be based "at least partly" on that M-bit precision data. Ex. 1003, ¶76.

[1f]/[7f]/[13f] [decreasing/decrease] a precision of said combined prediction by shifting bits of the combined prediction to the right; and

Ground 1 teaches [1f], [7f], and [13f]. Ex. 1003, ¶¶77-81.

Wada discloses decreasing the precision of the generated prediction image. Ex. 1004, ¶271. Specifically, the output of weighted prediction image generator 836 is N-bits (which matches the pixel bit depth of the decoded image) even though the input has a higher precision of L-bits.



F I G. 49

Ex. 1004, Fig. 49; *see also id.*, ¶271("[T]he processing is done so that the output of the pixel bit length of N-bit is obtained with respect to the input of the pixel bit length of L-bit.").

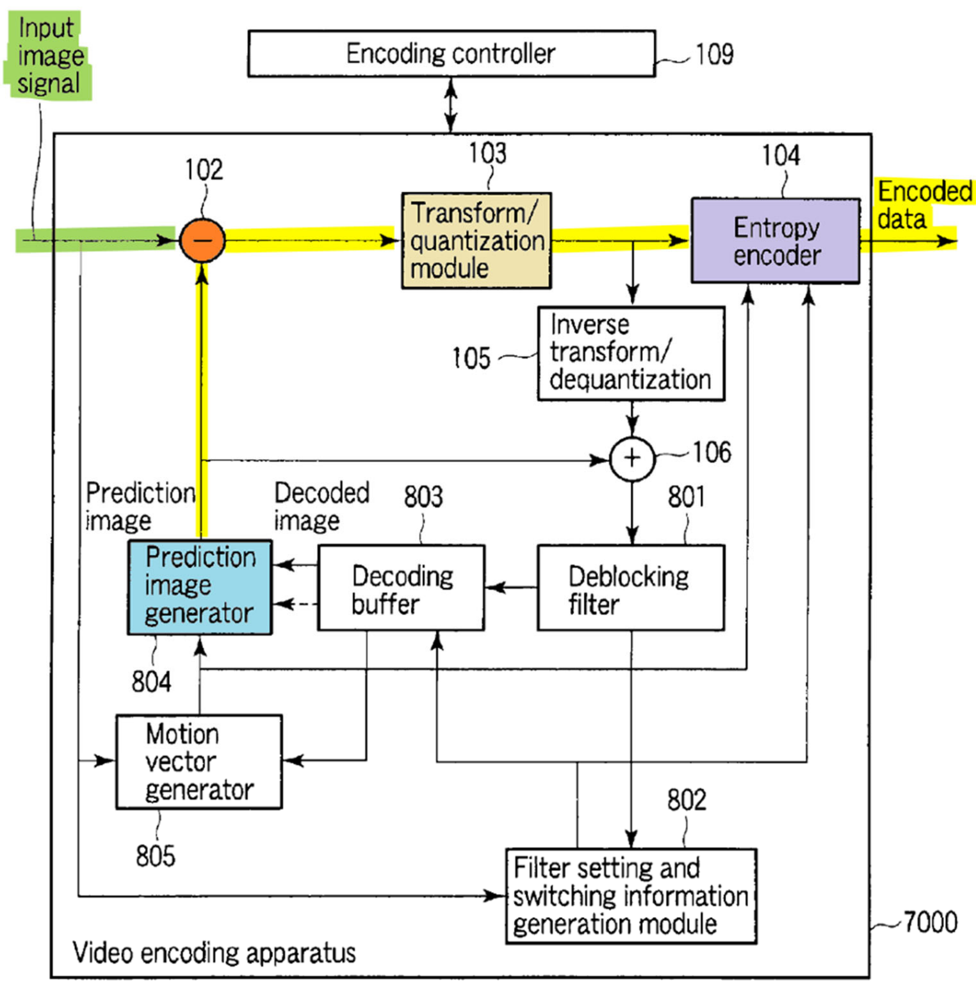
As discussed, it would have been obvious to a POSITA for Wada's prediction image generator to generate predictions for B video blocks in accordance with Karczewicz-I's AVC-based equations. *Supra* §V.A.3. Where, as Wada discloses, the image data input to the weighted prediction image generator has increased precision of L-bits as opposed to N-bits, the combined prediction obtained by averaging those values using Karczewicz-I's weighted prediction equations would have the same increased precision of L bits. Ex. 1003, ¶¶79-81.

It also would have been obvious to employ a right-shift to decrease the precision from L to N bits. First, a POSITA would have been familiar with using left- and right-shifts to adjust precision. *Supra* §I.C; Ex. 1003, ¶81. Second, Karczewicz-1 discloses using right-shifts to reduce the precision of interpolated subpixels. *E.g.*, Ex. 1006, 1 (interpolation equations using right-shifts by 6 bits); Ex. 1003, ¶81. Third, Wada's prediction image generator uses a left-shift to *increase* pixel bit depth. Ex. 1004, ¶270 (describing bit extension module 833). Using a right-shift to *decrease* precision would have been consistent with Wada's approach for increasing precision. Ex. 1003, ¶81. Fourth, Karczewicz-I's default sample weighted prediction formula already includes a right-shift. Ex. 1005, ¶60. It would have been obvious to increase the extent of that right shift (by L – N bits) to reduce the precision of the prediction image back to N bits as Wada discloses. Ex. 1003, ¶81.

[1g]/[7g]/[13g] [encoding/encode] residual data in a bitstream, wherein the residual data is determined based upon a difference between the combined prediction and the block of pixels.

Wada's encoding apparatus 7000 includes subtracter 102 which "generates a prediction error signal between the generated prediction image and the input image." Ex. 1004, ¶262. That "prediction error signal" reflects a difference between the actual block and the predicted block and is thus residual data. *Id.*, ¶79 ("The subtracter 102 calculates a *difference* between an acquired input image signal 10 and the prediction image signal 11 to output a prediction error image signal 12."). Ex. 1003, ¶83.

Fig. 46 shows that the residual output from subtracter 102 is transformed and quantized by block 103 and then entropy coded by block 104 and output in the encoded data:



F I G. 46

Id., Fig. 46; *see also id.*, ¶¶79, 89 (describing 103), ¶82 (describing 104), ¶100 (quantized transform coefficient 13 output from module 103 is multiplexed into bit stream transmitted to decoder), Fig. 1 (similar encoder with same components).

b. Dependent Claims 2, 8, and 14

2. The method according to claim 1, **8. The apparatus according to claim 7,** **14. The computer program product according to claim 13,** **wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.**

Ground 1 renders obvious claims 2, 8, and 14. Ex. 1003, ¶¶85-88.

Wada's prediction image generator 804 includes "interpolation picture generator 834 [which] generates an interpolation picture of a fractional pixel position as a prediction image, based on the motion vector." Ex. 1004, ¶¶268, 271.

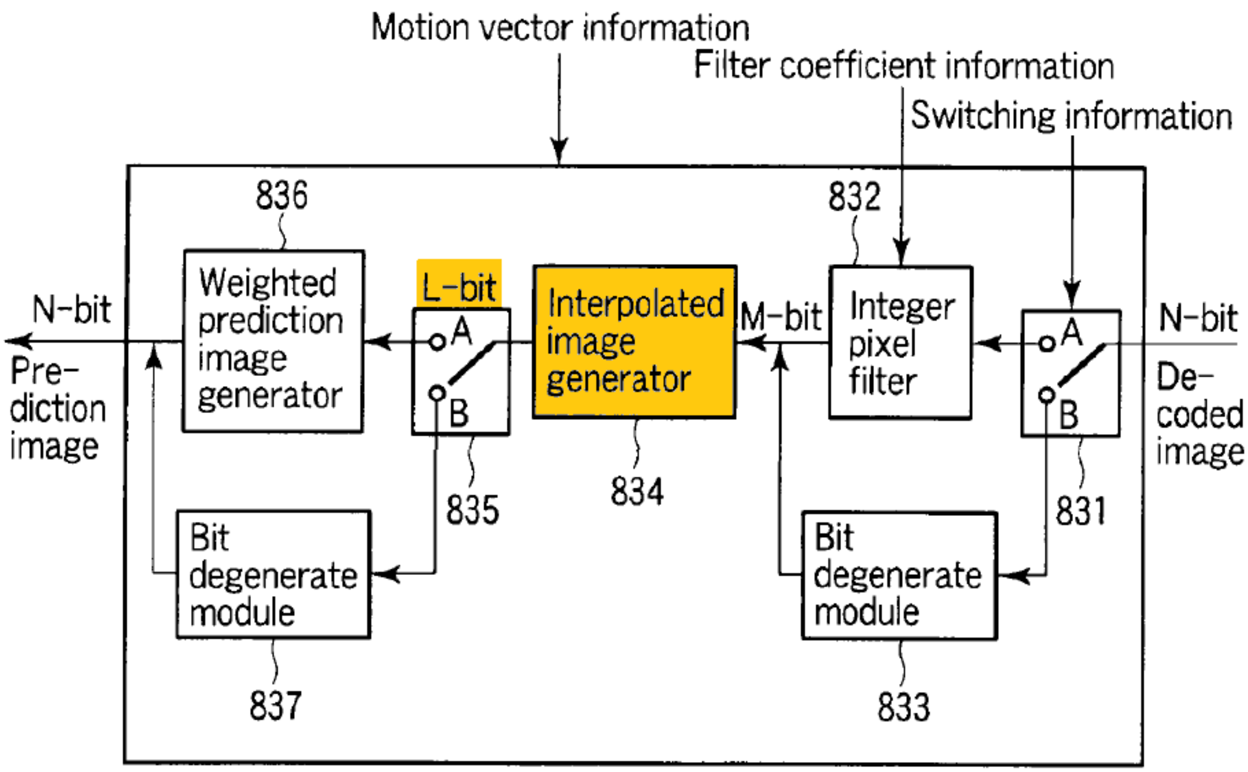


FIG. 49

Id., Fig. 49. As shown, the output of interpolated image generator 834 feeds into weighted prediction image generator 836. A POSITA would have understood that the reference blocks used to generate the weighted predictions would be interpolated where the motion vector points to a subpixel or fractional pixel location. Ex. 1003, ¶86. As noted, subpixel interpolation was well-understood by the time of the alleged invention. *Supra* §I.B.

Additionally, Karczewicz-I's AI33 submission discloses a "high precision" technique for interpolating pixels at half- and quarter-pixel positions. Ex. 1006.

The interpolation calculations use integer pixel values which are shaded and labeled in Figure 1 (below) with upper-case letters:

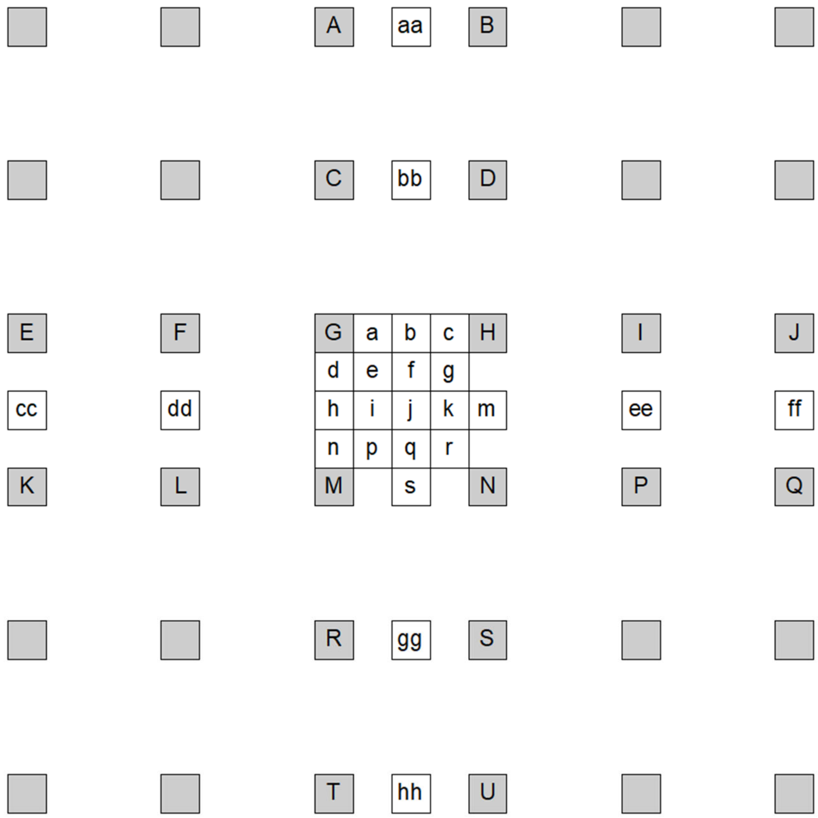


Figure 1: Integer pixels (shaded blocks with upper-case letters) and fractional pixel positions (white blocks with lower-case letters).

Id., Fig. 1; *see also id.*, 1 (interpolation equations).

In view of the above, it would have been obvious to a POSITA that where a motion vector points to a subpixel (as was well-known, *see supra* §I.B), the prediction would be obtained by interpolation using the block of integer pixel values from the reference image (*i.e.,* "pixel values of said first reference block") indicated by the motion vector. Ex. 1003, ¶88.

c. Dependent Claims 3, 9, and 15

3. The method according to claim 2,
9. The apparatus according to claim 8,
15. The computer program product according to claim 14,
wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.

(i) Ground 1

Ground 1 renders obvious claims 3, 9, and 15 for two reasons. Ex. 1003, ¶¶89-92.

First, Wada discloses that the input to interpolated image generator 834 is M-bits which is greater than N-bits. Ex. 1004, ¶270; Fig. 49. Karczewicz-I discloses that applying a 6-tap filter to 8-bit samples results in intermediate values for half pixels that require "15 bits plus sign (15s)." Ex. 1006, 3. It would have been obvious that applying a 6-tap filter to higher-precision samples (*i.e.*, M > 8 bits) would yield intermediate values for half-pixels requiring more bits and that this would have posed a problem for averaging with other samples using 16-bit arithmetic. Ex. 1003, ¶90. Karczewicz-I addresses a similar situation involving the half-pixel in the 'j' position where the intermediate value 'j1' generated from the P-tap filter requires 20 bits. Ex. 1006, 3. Karczewicz-I discloses right-shifting the sum of the P-tap filter to obtain a reduced precision, 16-bit representation of ‘j1’ which in turn is averaged with other intermediate values using 16-bit arithmetic:

r1 = r1 + r3	r1 is ((y0+y5)-5*(y1+y4)+20*(y2+y3))>>4	-9977	26232	16s
j1 = r1	**store result**			

Ex. 1006, 4. From these teachings, it would have been obvious to a POSITA to employ the same straightforward right-shift technique to intermediate values obtained from applying a P-tap filter to higher-resolution (M-bit) samples where, like Karczewicz-I’s ‘j1’ value, they can exceed 16 bits. Ex. 1003, ¶90.

Second, Karczewicz-I’s quarter-pixel interpolation calculations involve right-shifting a sum of a P-tap filter. For example, Karczewicz-I discloses interpolating quarter-pixel ‘a’ by summing integer pixel ‘G’ (left-shifted 5 bits) with ‘b1’ and a rounding adjustment of 32.

$$a = (G << 5 + b1 + 32) >> 6$$

Ex.1006 (AI33 submission), 1. This expression right-shifts the sum of a 2-tap filter. Ex. 1003, ¶91; Ex. 1019, ¶36 (in H.264, “[a]ll quarter-pixel locations undergo 2-tap average filtering using integer and half pixels”). Each quarter-pixel value in the first prediction would be computed using the above 2-tap filter based on integer pixels in location G from the reference picture which are values of the first reference block. Ex. 1003, ¶92. Additionally, b1 is “derived by applying [a] 6-tap filter … horizontally” using integer pixel values E, F, G, H, I, and J (which, again, are values of the first reference block). Ex. 1006, Fig. 1; Ex. 1003, ¶92.

(ii) Ground 1A

Ground 1A, which adds Srinivasan, also renders obvious claims 3, 9, and 15. Ex. 1003, ¶¶93-97. Srinivasan was published and publicly available no later than September 13, 2005 and qualifies as prior art under §102(b). Ex. 1026 (Hall-Ellis declaration).

Srinivasan describes how VC-1 performs sub-pixel interpolation in two dimensions. Ex. 1008, 000015. A POSITA would have known that sub-pixel interpolation in two dimensions is used to interpolate a subpixel in ½ pixel horizontal and ½ pixel vertical location (*e.g.*, the 'j' location in Karczewicz-I's AI33 submission). Ex. 1003, ¶95. "First, filtering is performed in the vertical direction" using a four-tap filter. Ex. 1008, 000015. Next, "a rounding factor is added to the result and *some bits are shifted out*"—in other words a right-shift is performed so that "the intermediate values … are within 16 bits." *Id.* Relatedly, Srinivasan explains that "[i]t is possible to achieve additional accuracy by retaining a higher precision result"—*i.e.,* higher than 8 bits—"after the first stage of filtering." *Id.* This first stage of Srinivasan's two-stage interpolation process right-shifts a sum of a P-tap filter using integer pixel values from the reference picture (*i.e.,* "using values of said first reference block") as claimed. Ex. 1003, ¶95. Then, the partially rounded intermediate results from the vertical filtering are filtered in the horizontal direction to obtain a result which is rounded and clipped to obtain the final interpolated value. Ex. 1008, 000015.

Srinivasan is analogous art in the same field as the ’267 patent (*supra* §III) because it is generally directed to video encoding/decoding and discusses motion compensation and subpixel interpolation. Ex. 1008, 000014-15; Ex. 1003, ¶**.

It would have been obvious to a POSITA to combine Srinivasan’s teachings regarding high-precision sub-pixel interpolation in two dimensions with the teachings of Wada and Karczewicz-I. As discussed for Ground 1, a POSITA would have understood that applying a P-tap filter to Wada’s higher-precision (M-bit) pixel values would result in intermediate values for interpolated half-pixels that exceed 16 bits. Srinivasan describes partially rounding intermediate values to allow for a 16-bit implementation while still obtaining the benefits of more accurate sub-pixel interpolation flowing from the increased precision of the intermediate values. Ex. 1008, 000015. Although Karczewicz-I’s AI33 submission discloses a similar technique, it would have been obvious to a POSITA to use Srinivasan’s approach in a given implementation, for example, because it uses a 4-tap filter rather than a 6-tap filter which would have reduced the number of calculations during interpolation. Ex.. 1003, ¶96.

A POSITA would have had a reasonable expectation of success in combining the teachings of Srinivasan with those of Wada and Karczewicz-I. Ex.

1003, ¶97. Srinivasan's technique of right-shifting intermediate values to reduce them to 16-bit precision is straightforward and it would have been well-within the level of ordinary skill to apply similar right shifts to intermediate values obtained from the sum of a P-tap filter applied to Wada's higher-resolution (M-bit) samples. *Id.* Although Srinivasan describes VC-1, Petitioners do not propose to incorporate any aspect of VC-1 beyond Srinivasan's disclosure of high-precision sub-pixel interpolation.

d. Dependent Claims 4, 10, and 16

4. The method according to claim 2,
10. The apparatus according to claim 8,
16. The computer program product according to claim 14,
wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

Ground 1 renders obvious claims 4, 10, and 16. Ex. 1003, ¶¶98-101.

First, Wada discloses that prediction image generator 804 includes bit extension module 833 which left-shifts values from the decoded image to increase their pixel bit depth from N to M bits. Ex. 1005, ¶270, Fig. 49. The M-bit samples are fed into interpolated image generator 834. *Id.*, Fig. 49. It would have been obvious to a POSITA that, where the motion vector pointed to reference pixels at integer positions, the prediction would not rely on interpolated subpixels but simply use the integer pixel values from the decoded image that were increased from N to M bits by bit extension module 833. Ex. 1003, ¶99.

Second, Karczewicz-I discloses left-shifting integer pixel values (*e.g.,* G, H, M) before averaging them with 15-bit interpolated half-pixel values (*e.g.,* b1, h1).

```
a = (G<<5+b1+32)>>6
c = (H<<5+b1+32)>>6
d = (G<<5+h1+32)>>6
n = (M<<5+h1+32)>>6
```

Ex. 1006, 1; *id.*, 3 (noting b1 and h1 require 15 bits). Because, as discussed, biprediction involves averaging pixels that can be represented with varying precision, it would have been obvious to a POSITA to left-shift the integer pixel values to match the higher precision intermediate values for interpolated subpixel values in the same way Karczewicz-I describes. Ex. 1003, ¶100.

The above teachings, taken separately or in combination, render obvious claims 4, 10, and 16. Ex. 1003, ¶101.

e. Dependent Claims 5, 11, and 17

5. The method according to claim 1, wherein said decreasing said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprises:

11. The apparatus according to claim 7, wherein the at least one memory and computer code are configured to cause the apparatus to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, by:

17. The computer program product according to claim 13, wherein the program code instructions configured to decrease said precision of said combined prediction by shifting bits of the combined prediction

to the right, further comprise program code instructions configured to:

[inserting/insert] a rounding offset to the combined prediction before said decreasing.

Ground 1 renders obvious claims 5, 11, and 17. Ex. 1003, ¶¶102-103.

Karczewicz-I discloses adding a rounding offset to the combined prediction before the right shift in computing a bidirectional default weighted prediction:

$$\text{pred}(i,j) = (\text{pred0}(i,j) + \text{pred1}(i,j) + 1) >> 1$$

Ex. 1005, ¶60; *see also id.*, ¶55 (explaining that weighted prediction in AVC involves adding a "rounding adjustment" of 1 "to the addition of the weighted blocks of the first and second frames and then shifting the result to the right by one bit"). As discussed for limitations [1f]/[7f]/[13f], it would have been obvious to a POSITA to increase the right shift by M – N bits in Karczewicz-I's weighted prediction calculations where higher precision sample values are used as Wada discloses. *Supra* §V.A.4.a; *see also* Ex. 1005, ¶63 ("Generally, a rounding adjustment of 2^{r-1} is commonly used prior to a right shift by r, where r represents a positive integer."). These rounding adjustments are rounding offsets as claimed. Ex. 1003, ¶102.

f. Dependent Claims 6, 12, and 18

6. The method according to claim 1,

12. The apparatus according to claim 7,

18. The computer program product according to claim 13,

wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

Ground 1 renders obvious claims 6, 12, and 18. Ex. 1003, ¶¶104-106.

As I discussed for claims 1, 7, and 13, the "first precision" in Wada is N bits which is the number of bits needed to represent the values of the pixels in the decoded images. Ex. 1004, ¶270 ("the pixel bit length of the decoded image is N bits"); Fig. 49. Ex. 1003, ¶104.

As also discussed, the claimed "first prediction" and "second prediction" could be the L-bit prediction data output from Wada's interpolated image generator 834 and the "second precision" would be L bits. A POSITA would have understood L bits are needed to represent the output of interpolated image generator 834. Ex. 1004, ¶271 (the input to interpolation picture generator 834 "is the pixel bit length of M-bit whereas the output is the pixel bit length of L-bit wherein L≧N"); Ex. 1003, ¶105. Further, as discussed, Karczewicz-I shows how subpixel interpolation can generate higher-precision intermediate values for interpolated subpixels. Ex. 1006.

Alternatively, as also discussed for claims 1, 7, and 13, the claimed predictions could be the M-bit output of integer pixel filter 832 and/or bit extension module 833 in which case the "second precision" would be M bits. Wada suggests M bits are needed to represent these values. Ex. 1004, ¶270. Ex. 1003, ¶106.

5. Decoder Claims

The decoder-side claims of the ’267 patent recite almost identical limitations to the encoder-side claims discussed above. Moreover, Wada discloses that the same prediction image generator depicted in Fig. 49—which was the focus of the discussion above for the encoder-side claims—is used both for encoding and decoding. Ex. 1004, ¶269. Similarly, Karczewicz-I discloses that the decoder performs “the reciprocal decoding techniques to the encoding techniques described” for the encoder including support for weighted prediction. Ex. 1005, ¶83, *see also id.*, ¶¶84-86, 88-89 (discussing decoding including block based predictive decoding using weighted prediction data); ¶44 (“For different types of weighted bi-directional prediction in accordance with ITU-T H.264, … video *decoder* 28 may generally support three different types of prediction modes.”). Ex. 1003, ¶107.

Accordingly, the discussion below for the decoder-side claims appropriately refers back to the detailed discussion above of identical or nearly identical limitations of the encoder-side claims.

a. Independent Claims 19, 25, and 31

[19a]. A method for decoding a block of pixels, the method comprising:

Ground 1 teaches 19[a]. Ex. 1003, ¶¶109-111.

Wada teaches a method for decoding blocks of pixels, for example, in connection with the decoding apparatus 8000 shown in Fig. 47. Ex. 1004, ¶¶263-266. Wada's decoding method uses a prediction image generator that generates motion-compensated prediction images. *Id.*, ¶266. Wada discloses block-based coding methods. *Id.*, ¶102 (describing "a macroblock level syntax (1907) [that] recites transform coefficient data, prediction mode information and a motion vector which are needed for every macroblock"). *Id.*, ¶102, Fig. 31.

Similar to Wada, Karczewicz-I describes decoding methods that use bidirectional prediction to decode blocks within video frames. Ex. 1005, ¶8 ("In bi-directional prediction, a *video block* may be predictively encoded and decoded based on two different lists of predictive reference data."); *id.*, ¶39 ("Video encoder 22 and video decoder 28 may operate on *video blocks* within individual video frames in order to encode and decode the video data.").

[25a]. An apparatus for decoding a block of pixels, the apparatus comprising: at least one processor and at least one memory including computer program code, the at least one memory and computer program code configured to, with the at least one processor, cause the apparatus to:

Ground 1 teaches 25[a]. Ex. 1003, ¶¶112-113.

Wada discloses video decoding apparatus 2000 which "can be realized using general-purpose computer equipment." Ex. 1004, ¶132. Wada further explains that the various decoder blocks "can be realized by making a processor mounted on the computer equipment execute a program" after, for example, "installing the program in the computer equipment beforehand." *Id.* By the same token, a POSITA would have understood from Wada that video decoding apparatus 8000 of Fig. 47 could likewise be implemented with conventional computer technology. Ex. 1003, ¶113. By the time of the alleged invention, a POSITA would have been well-aware that computers with processors, memory, and program code were routinely used to decode video. Ex. 1003, ¶113. Similar to Wada, Karczewicz-I discloses that the described decoding techniques can be implemented using processors and software stored in memory. Ex. 1005, ¶¶98-101.

[31a]. A computer program product for decoding a block of pixels, the computer program product comprising at least one non-transitory computer readable storage medium having computer executable program code portions stored therein, the computer executable program code portions comprising program code instructions configured to:

As discussed for preamble [25a], both Wada and Karczewicz-I teach computer-based decoders that execute installed software. Ex. 1003, ¶114. Karczewicz-I specifically describes "a computer-readable medium comprising instructions that, when executed in a processor, performs" the described decoding methods. Ex. 1005, ¶99.

[19b]/[25b]/[31b] [determining/determine], for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

Limitations [19b], [25b], and [31b] are identical to limitations [1b], [7b], and [13b]. Ground 1 teaches or suggests these limitations as discussed. *Supra* §V.A.4.a. Ex. 1003, ¶116.

[19c]/[25c]/[31c] [using/use] said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

Limitations [19c], [25c], and [31c] are identical to limitations [1c], [7c], and [13c]. Ground 1 teaches or suggests these limitations as discussed. *Supra* §V.A.4.a. Ex. 1003, ¶117.

[19d]/[25d]/[31d] [using/use] said second reference block to obtain a second prediction, said second prediction having the second precision;

Limitations [19d], [25d], and [31d] are identical to limitations [1d], [7d], and [13d]. Ground 1 teaches or suggests these limitations as discussed. *Supra* §V.A.4.a. Ex. 1003, ¶117.

[19e]/[25e]/[31e] [obtaining/obtain] a combined prediction based at least partly upon said first prediction and said second prediction;

Limitations [19e], [25e], and [31e] are identical to limitations [1e], [7e], and [13e]. Ground 1 teaches or suggests these limitations as discussed. *Supra* §V.A.4.a. Ex. 1003, ¶118.

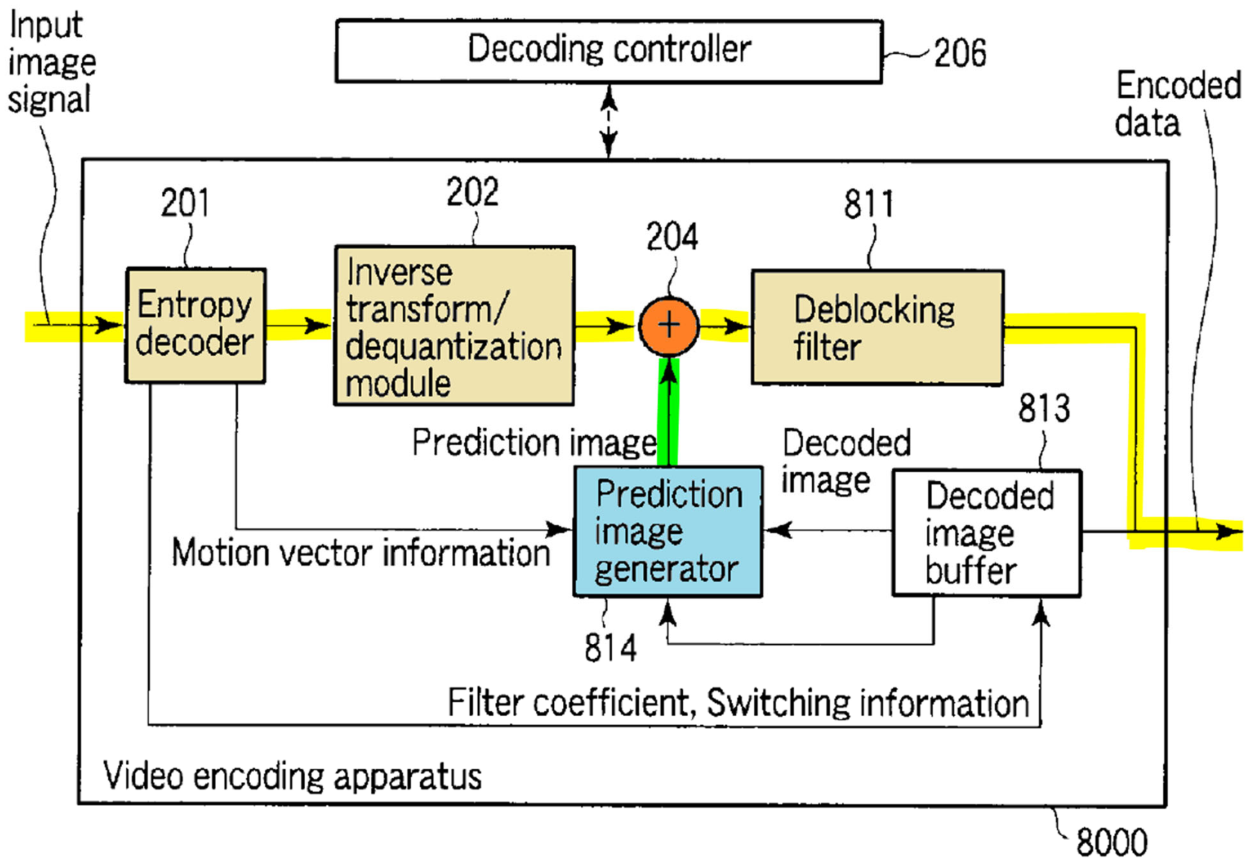
[19f]/[25f]/[31f] [decreasing/decrease] a precision of said combined prediction by shifting bits of the combined prediction to the right; and

Limitations [19f], [25f], and [31f] are identical to limitations [1f], [7f], and [13f]. Ground 1 teaches or suggests these limitations as discussed. *Supra* §V.A.4.a. Ex. 1003, ¶119.

[19g]/[25g]/[31g] [reconstructing/reconstruct] the block of pixels based on the combined prediction.

Wada's decoding apparatus 8000 includes subtracter 102 which "generates a prediction error signal between the generated prediction image and the input image." Ex. 1004, ¶262. That "prediction error signal" reflects a difference between the actual block and the predicted block and is thus residual data. *Id.,* ¶79 ("The subtracter 102 calculates a *difference* between an acquired input image signal 10 and the prediction image signal 11 to output a prediction error image signal 12."). Ex. 1003, ¶120.

Fig. 47 shows that the output of prediction image generator—the combined prediction where the current picture is a B-picture—is added by 204 with decoded residual, deblocked, and output:



F I G. 47

Id., Fig. 45;[5] *see also id.*, ¶¶263-265; *see also id.*, ¶¶110, 120 ("The prediction error image signal 15 is added to the prediction image signal 11 output from the prediction signal generator 203 with the adder 204, whereby the decoded image signal 21 is generated."), Fig. 5 (similar decoder with same components).

---

[5] Fig. 47 includes readily apparent typos (e.g., "video ***en***coding apparatus") and a POSITA would have understood that, consistent with Fig. 5, the input to decoding apparatus 8000 would be encoded data and the output would be an output image signal (*i.e.,* the reconstructed image). Ex. 1003, ¶121.

b. Dependent Claims 20, 26, and 32

20. The method according to claim 19, **26. The apparatus according to claim 25,** **32. The computer program product according to claim 31,** **wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.**

These claims add the same limitations as encoder-side dependent claims 2, 8, and 14 and are obvious for the same reasons. *Supra* §V.A.4.b. Ex. 1003, ¶122.

c. Dependent Claims 21, 27, and 33

21. The method according to claim 20, **27. The apparatus according to claim 26,** **33. The computer program product according to claim 32,** **wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.**

These claims add the same limitations as encoder-side dependent claims 3, 9, and 15 and Grounds 1 and 1A render them obvious for the same reasons. *Supra* §V.A.4.c. Ex. 1003, ¶123.

d. Dependent Claims 22, 28, and 34

22. The method according to claim 20, **28. The apparatus according to claim 26,** **34. The computer program product according to claim 32,**

wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

These claims add the same limitations as encoder-side dependent claims 4, 10, and 16 and are obvious for the same reasons. *Supra* §V.A.4.d. Ex. 1003, ¶124.

e. Dependent Claims 23, 29, and 35

23. The method according to claim 19, wherein said decreasing said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprises:

29. The apparatus according to claim 25, wherein the at least one memory and computer code are configured to cause the apparatus to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, by:

35. The computer program product according to claim 31, wherein the program code instructions configured to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprise program code instructions configured to:

[inserting/insert] a rounding offset to the combined prediction before said decreasing.

These claims add the same limitations as encoder-side dependent claims 5, 11, and 17 and are obvious for the same reasons. *Supra* §V.A.4.e. Ex. 1003, ¶125.

f. Dependent Claims 24, 30, and 36

24. The method according to claim 19,

30. The apparatus according to claim 25,

36. The computer program product according to claim 31,

wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the

number of bits needed to represent values of said first prediction and values of said second prediction.

These claims add the same limitations as encoder-side dependent claims 6, 12, and 18 and are obvious for the same reasons. *Supra* §V.A.4.f. Ex. 1003, ¶126.

B. Grounds 2 & 2A

1. Karczewicz-I (Ex. 1005)

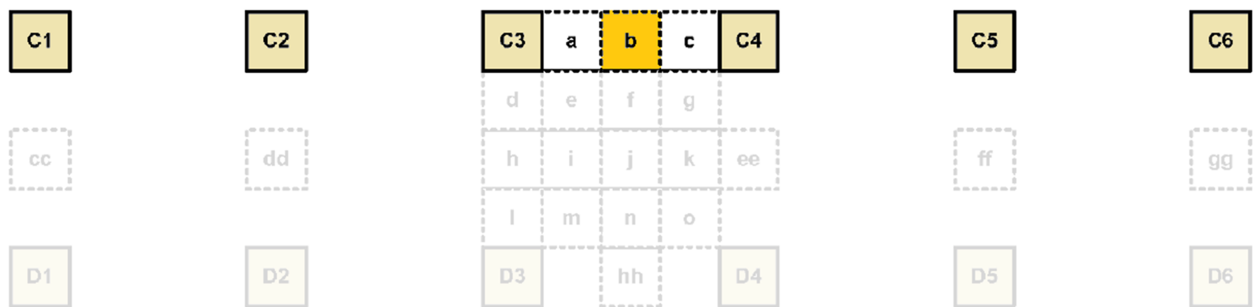
As discussed for Ground 1, Karczewicz-I discloses coding techniques related to bidirectional prediction. *Supra* § V.A.2. Ex. 1003, ¶130.

2. Karczewicz-II (Ex. 1007)

Karczewicz-II is U.S. Patent Application Publication 2009/0257499, published October 2009; it is prior art under §§102(a), (b) and (e). Ex. 1003, ¶133.

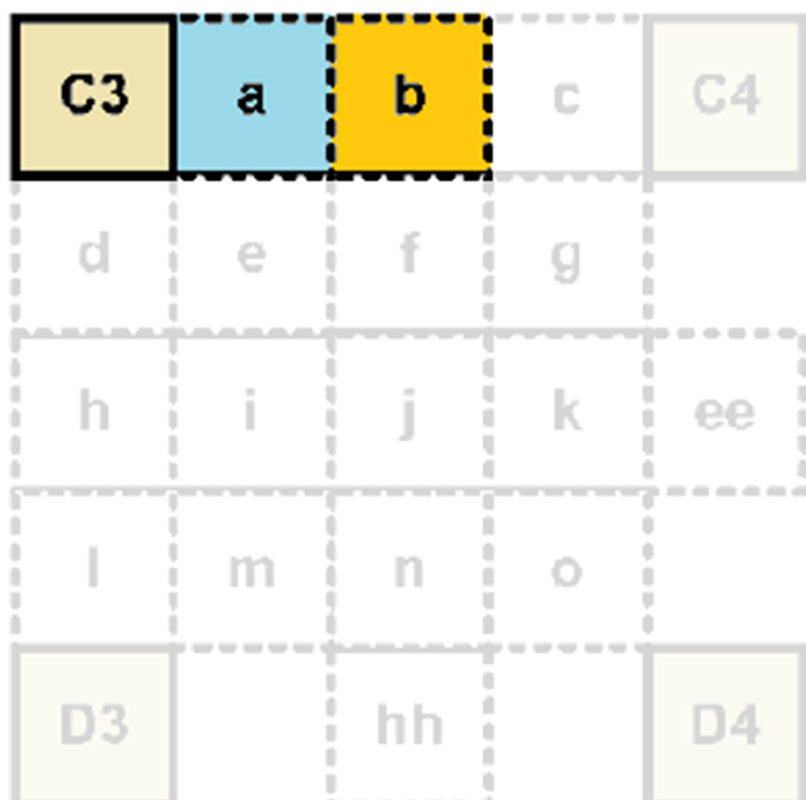
Karczewicz-II explains that the H.264 standard approach for quarter-pixel interpolation causes "coding inefficiencies" due to its use of rounded half-pixel values. Ex. 1007, ¶¶10, 39. Karczewicz-II discloses an improved approach that uses "non-rounded versions of the half-pixel values" for quarter-pixel interpolation. *Id.* This approach "can eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values." *Id.*

First, half-pixels are interpolated consistent with H.264. In the figure below, half-pixel 'b' is interpolated using the integer pixels 'C1' through 'C6' in the same row:



Id., Fig. 4B (excerpted). Initially, an ***intermediate value 'b1'*** is calculated. *Id.*, ¶93. The value of 'b1' is between -2550 and 10,710. *Id.*, ¶103 (Table 5). Then 'b1' is right-shifted and clipped to an ***8-bit value 'b'***. *Id.*, ¶94.

The H.264 standard used ***rounded*** half-pixels to interpolate quarter-pixels. For example, in the figure below, quarter-pixel 'a' is interpolated using rounded half-pixel 'b' and integer pixel 'C3.'



Ex. 1007, Fig. 4A (excerpted, annotations added). Conventionally, the rounded (*i.e.*, 8-bit) value 'b' is used:

$$a = (C3 + b + 1) >> 1$$

Ex. 1007, ¶96 (Table 1).

Rather than the final, rounded value 'b', Karczewicz-II teaches using the higher-precision, non-rounded value 'b1' during quarter-pixel interpolation:

$$a = (C3 << 5 + b1 + 32) >> 6$$

Ex. 1005, ¶99 (Table 3). Notably, Karczewicz-II's modified equation (i) left-shifts the integer pixel 'C3' to increase its precision to match that of 'b1', (ii) increases the rounding adjustment from 1 to 32,[6] and (iii) increases the extent of the right-shift to round the quarter-pixel result back to standard, 8-bit precision. Ex. 1003, ¶154.

Karczewicz-II provides equations for interpolating quarter-pixels in each position. One set of equations interpolates quarter-pixels 'a', 'c', 'd', 'l' using one integer pixel and one non-rounded half-pixel. Ex. 1005, ¶99 (Table 3); ¶103 (Table 5). Another set of equations interpolates quarter-pixels 'e', 'g', 'm', and 'o' using two non-rounded half-pixels. *Id.*, ¶99 (Table 3), ¶103 (Table 6). And another set of equations interpolates quarter-pixels 'f', 'i', 'k', and 'n' using one

[6] As Karczewicz-I explains, "a rounding adjustment of 2^{r-1} is commonly used prior to a right shift by r, where r represents a positive integer." Ex. 1005, ¶63. Here, the right shift is by 6 and the rounding adjustment is thus 2^5 or 32. Ex. 1003, ¶154.

non-rounded half-pixel and a partially rounded half-pixel j1. *Id.*, ¶101 (Table 4), ¶105 (Table 8). The different equations use different shifts depending on the precision of the pixels used to interpolate the subpixel. Ex. 1003, ¶154. For example, integer pixel values are left shifted to match the precision of the non-rounded half-pixels prior to averaging. Ex. 1007, ¶99 (Table 3 calculations for a/c/d/l). The special case j1 half-pixel (which has a higher precision owing to how it is calculated) is right shifted to match the precision of the non-rounded half pixels before averaging. Ex. 1007, ¶99 (Table 3 calculations for f/i/k/n).

Notably, Karczewicz-II's quarter-pixel interpolation calculations are the same calculations used in certain embodiments of the '267 patent which likewise perform two half-pixel interpolations (one for each reference block) and average the higher-precision intermediate values for the half-pixels together to interpolate the quarter-pixels. Ex. 1001, 12:41-13:55.

Karczewicz-II is analogous art in the same field as the '267 patent (*supra* §III) because it is directed to video encoding/decoding and H.264. E.g., Ex. 1007, ¶2, ¶¶46-47; Ex. 1003, ¶136. Like the '267 patent, Karczewicz-II was assigned the H04N subclass. Ex. 1007, cover.

3. Motivation to Combine

Karczewicz-I and Karczewicz-II are Qualcomm patent applications having the same inventors. Both apply their teachings to similar video-coding architectures. Ex. 1005, ¶2, ¶¶29-50, Fig. 1; Ex. 1007, ¶2, ¶¶40-53, Fig. 1. Both are directed to block-based H.264 inter prediction. Ex. 1005, ¶35, ¶44, ¶¶55-60; Ex. 1007, ¶8, ¶¶35-36, ¶46, ¶54. Both teach encoders having similar architectures that perform motion estimation and motion compensation for inter-predictive coding:

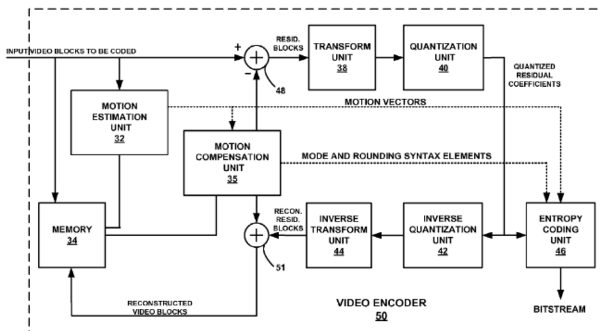


FIG. 2

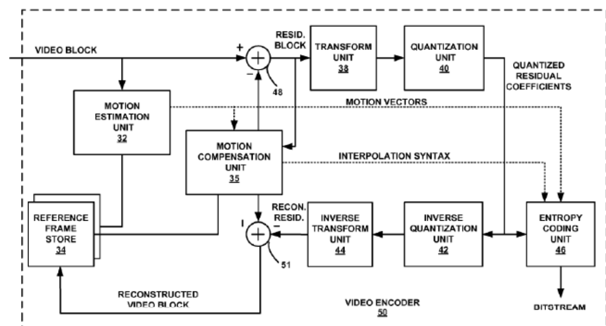


FIG. 2

Ex. 1005, Fig. 2 (above left), ¶53; Ex. 1007, Fig. 2 (above right), ¶56. Ex. 1003, ¶138. Both teach decoders having similar architectures that perform motion compensation for block-based decoding:

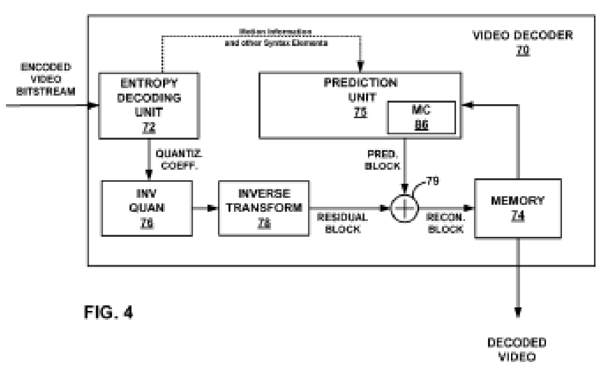


FIG. 4

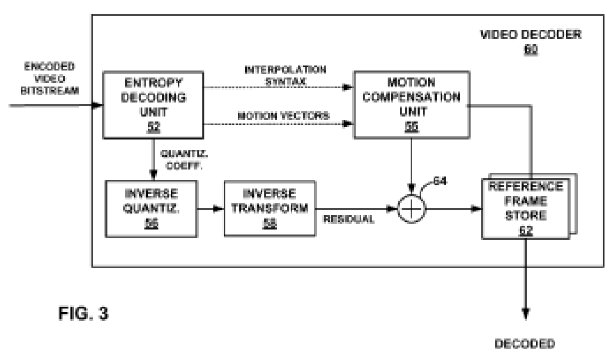


FIG. 3

Ex. 1005, Fig. 4 (above left); ¶89; Ex. 1007, Fig. 3 (above right), ¶63. Ex. 1003, ¶139. Karczewicz-I incorporates by reference a publication that describes a "High

Precision Interpolation" approach similar to the approach described in Karczewicz-II. Ex. 1007. Ex. 1003, ¶48.

Both references disclose motion compensation techniques that generate predictions by averaging pixel values. Karczewicz-I's motion compensation unit 35 generates predictions for B-video blocks by averaging pixels from different reference pictures based on motion vectors determined by motion estimation. Ex. 1005, ¶¶43-46, 54, 60-64, 78. Karczewicz-II's own motion compensation unit 35 "generate[s] predictive data at sub-pixel resolution" based on motion vectors determined by motion estimation. Ex. 1007, ¶57; *id.*, ¶¶96-102 (describing interpolation of quarter-pixels by averaging half-pixels with other half-pixels or with integer pixels). Further, Karczewicz-I suggests—and a POSITA would have understood—that H.264 bidirectional prediction may rely on interpolated sub-pixels. Ex. 1005, ¶37 (incorporating by reference AI33 submission describing high precision interpolation); 41 ("inter-based predictive encoding … includes interpolation"); *see also* Ex. 1001, 2:60-3:11 (discussing, as background, interpolation of half- and quarter-pixel samples in H.264); Ex. 1003, ¶81.

More broadly, as discussed, bidirectional prediction is a type of interpolation. *Supra* §I.A. Karczewicz-I interpolates prediction values using pixel values from two different reference pictures. Karczewicz-II interpolates sub-pixel prediction values using different pixel values from the same picture. Ex. 1003, ¶134.

It would have been obvious to a POSITA to improve Karczewicz-I's H.264 default weighted prediction calculations in the same way Karczewicz-II improves the ***highly similar*** H.264 quarter-pixel interpolation calculations. Both H.264 calculations average two pixel values by summing them (orange), adding a rounding adjustment of 1 (blue), and right-shifting the sum by 1 bit (purple):

$$\begin{array}{lllll} \mathrm{a} & = (\mathrm{C3} & + \mathrm{b} & + 1) & \gg 1 \\ \mathrm{pred}(i,j) & = (\mathrm{pred0}(i,j) & + \mathrm{pred1}(i,j) & + 1) & \gg 1 \end{array}$$

Ex. 1007, ¶96 (Table 1) (top); Ex. 1005, ¶60 (bottom). Moreover, as noted, the inputs to Karczewicz-I's default weighted prediction equation (*i.e.*, pred0(*i,j*) and pred1(*i,j*)) can be interpolated subpixels. Accordingly, the use of rounded half-pixel values in Karczewicz-I's default weighted prediction equation raises the same concern about "propagation of rounding inaccuracies" that Karczewicz-II addresses by using higher-precision intermediate values. Ex. 1007, ¶¶10,39. Ex. 1003, ¶135.

Relatedly, a POSITA would have recognized that Karczewicz-II teaches *precisely* how to modify Karczewicz-I's default weighted prediction equation to use higher-precision, non-rounded subpixel values in different scenarios.

Karczewicz-II discloses high-precision equations for averaging three different combinations of pixel types:

Scenario	Pixel Types	High-Precision Interpolation Equations
1	half & integer	*See* Tables 3 & 5 (a, c, d, l) Example: a = (b1 + C3 << 5 + 32) >> 6
2	center & half	*See* Tables 3 & 8 (f, I, k, n) Example: f = (j1 >> 5 + b1 + 32) >> 6
3	half & half	*See* Tables 4 & 6 (e, g, m, o) Example: e = (e1 + h1 + 32) >> 6

Ex. 1003, ¶¶150-151. Karczewicz-I's biprediction equations would have been used to average the same combinations of different pixel types and those calculations could be improved to use higher-precision pixel values in precisely the same way:

Scenario	Pixel Types	High-Precision Biprediction Equation
1	half & integer	pred(*i*,*j*) = (non-rounded(pred0(*i*,*j*)) + pred1(*i*,*j*)<<5 + 32) >> 6
2	center & half	pred(*i*,*j*) = (non-rounded(pred0(*i*,*j*))>>5 + non-rounded(pred1(*i*,*j*)) +32) >> 6
3	half & half	pred(i,j) = (non-rounded(pred0(i,j)) + non-rounded(pred1(i,j)) + 32) >> 6

Ex. 1003, ¶¶150-151. As Dr. Freedman further explains, it would have been straightforward for a POSITA to extend Karczewicz-II's teachings to perform high-precision averaging to other scenarios such as those involving quarter-pixels. *Id.*, ¶151.

Karczewicz-II teaches that "[b]y preserving the full precision of the intermediate values, the interpolated sub-pixels will be *more accurate*." Ex. 1007, ¶102. A POSITA would have readily understood that the use of higher precision intermediate values when interpolating bidirectional predictive data as described in would similarly provide for "more accurate" predictions for B-video blocks. Ex. 1003, ¶148. *See KSR v. Teleflex*, 550 U.S. 398, 401 (2007) ("[I]f a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond that person's skill."). Ex. 1003, ¶142.

As discussed, the benefits of using higher-precision intermediate values for video-coding computations were well-known in the art. *Supra* §I.C. Ex. 1003, ¶171. Using higher-precision intermediate values was known to improve motion compensation and weighted prediction. Ex. 1009, 7:4-19 ("a prediction and a reconstruction for a block of a picture to be predicted ... *in a higher precision*."), 16:22-17:36; Ex. 1023, 1 (noting that "weighted prediction in the video codec would depend on the arithmetic precision" and that "generally, ideal results are achieved by increasing arithmetic precision"). This approach was known to reduce the size of residual information. Ex. 1009, 7:4-19; 16:22-17:36; Ex. 1003, ¶171. It was also known that preserving higher-precision intermediate values improved the accuracy of interpolated pixels. Ex. 1008, 000015 ("possible to achieve additional accuracy [in sub-pixel interpolation] by retaining a higher precision result after the first stage of filtering"); Ex. 1010, ¶¶61-62; Ex. 1003, ¶172.

Compatible Teachings. The Ground 2 combination would not have changed the principle of operation of either reference, but merely includes the use of Karczewicz-II's known techniques to improve Karczewicz-I's similar devices and methods in the same way. Ex. 1005, Figs. 1-2, ¶¶29-50, ¶53; Ex. 1007, Figs. 1-2, ¶¶40-53, ¶56. Given the similarities between Karczewicz-I and Karczewicz-II, a POSITA would have understood that Karczewicz-II's techniques are readily applicable to Karczewicz-I. Ex. 1003, ¶173.

Moreover, the combination merely changes when rounding occurs for calculations that already included rounding. *See* Ex. 1005, ¶60, ¶55; Ex. 1007, ¶¶96-106, Tables 1-8. This minor implementation detail would not have changed the principle of operation of Karczewicz-I or Karczewicz-II. Ex. 1003, ¶174.

Reasonable Expectation of Success. A POSITA would have had a reasonable expectation of success in combining the teachings of Karczewicz-I and

Karczewicz-II as disclosed. As noted, the references disclose similar encoder and decoder architectures. In Karczewicz-II, the block that uses the higher-precision half-pixel values to interpolate quarter-pixel predictions is motion compensation unit 35. In Karczewicz-I, it is also motion compensation unit 35 that generates the weighted prediction. Accordingly, a POSITA would have recognized that Karczewicz-I's motion compensation unit 35 could be made to perform Karczewicz's II's improved calculations to generate more accurate bidirectional predictive data.

Additionally, a POSITA would have had a reasonable expectation of success because the combination applies the math behind Karczewicz-II's improved calculations to Karczewicz-I's scenarios without further modification. Ex. 1003, ¶175.

Furthermore, a POSITA would have been more than capable of applying Karczewicz-II's teachings because Karczewicz-II's calculations involve basic mathematic and logical operations (e.g., addition and bit-shifting) that were well known. *Supra* §I.C; Ex. 1003, ¶176.

The motivation for additionally combining Srinivasan for Ground 2A is discussed below in addressing claims 3, 9, and 15. *Infra* §V.B.4.c.

4. Encoder Claims

a. Independent Claims 1, 7, and 13

Preamble [1a]

Ground 2 teaches [1a]. Ex. 1003, ¶¶177-183.

Karczewicz-I "relates to … video encoding techniques that use bi-directional prediction" including for encoders. Ex. 1005, ¶2, ¶¶8-9, ¶22, ¶¶29-30, ¶32, Fig. 1. The encoder "perform[s] … inter-coding of blocks within video frames" that includes motion estimation and compensation operations. Ex. 1005, ¶¶51-76, Fig. 2; Ex. 1003, ¶178. Karczewicz-I encodes "blocks of pixel data[.]" *See, e.g.*, Ex. 1005, ¶¶39-40; Ex. 1003, ¶179.

Karczewicz-II "describes various interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding." Ex. 1007, Abstract, ¶2, ¶12, ¶4, ¶¶40-41, ¶43, ¶¶54-62, Figs. 1-2. Karczewicz-II encodes a block of pixels. Ex. 1007, ¶49, ¶50; Ex. 1003, ¶¶180-181.

It was obvious to modify Karczewicz-I's encoding method, based on Karczewicz-II's interpolation techniques. *Supra* §V.B.3 (explaining combination). The combination teaches a method for encoding a block of pixels, comprising the operations explained below for [1b]-[1g]. Ex. 1003, ¶183.

Preamble [7a]

Ground 2 teaches [7a]. Ex. 1003, ¶¶184-191.

Ground 2 teaches a video encoder for encoding a block of pixels that performs operations described below for [7b]-[7g]. *Supra* §V.B.4.a[1a] (explaining encoder); Ex. 1005, ¶¶29-30, ¶32, Fig. 1, ¶¶51-76, Fig. 2:

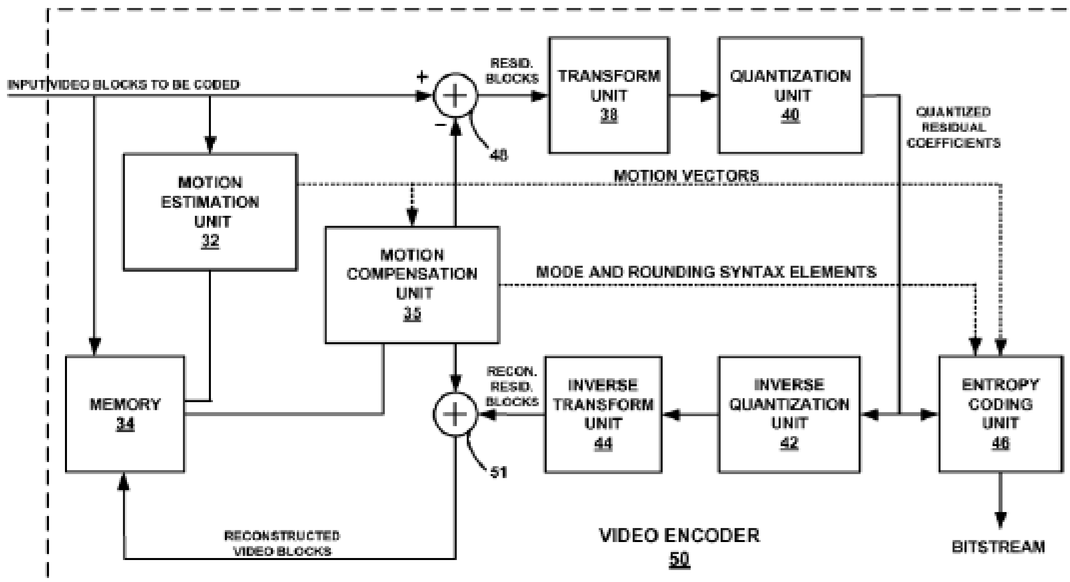


FIG. 2

Ex. 1007, ¶¶40-41, ¶43, Fig. 1, ¶¶54-62, Fig. 2; Ex. 1003, ¶186.

The video encoder encodes a block of pixels. Ex. 1005, ¶¶39-40; Ex. 1007, ¶¶49-50. Karczewicz-I and Karczewicz-II implement their encoder teachings in various types of devices (Ex. 1005, ¶3; Ex. 1007, ¶3) using a processor and memory (computer readable medium) that includes software, which comprises computer program code. Ex. 1005, ¶12, ¶38, ¶¶98-100; Ex. 1007, ¶¶25-27, ¶¶118-120. The medium comprises well-known memory types (e.g., RAM, ROM). Ex. 1005, ¶99; Ex. 1007, ¶119; Ex. 1003, ¶¶187-189.

Karczewicz-I and Karczewicz-II teach a processor executing computer program code in memory to carry out their teachings. Ex. 1005, ¶¶99-100; Ex. 1007, ¶48, ¶¶119-120. The apparatus further performs operations described below for [7b]-[7g]. Ex. 1003, ¶191.

Preamble [13a]

Ground 2 teaches the preamble. Ex. 1003, ¶¶192-194.

As explained above, Karczewicz-I and Karczewicz-II implement their video-encoder teachings and perform operations for encoding blocks of pixels using software stored in computer-readable storage media and executed by a processor. *Supra* §V.B.4[7a]; Ex. 1005, ¶12, ¶38, ¶¶98-100; Ex. 1007, ¶¶25-26, ¶48, ¶¶119-120. These types of media (e.g., RAM, ROM) are non-transitory. Ex. 1005, ¶99; Ex. 1007, ¶119. The software includes computer-executable program code portions that comprise program code instructions. Ex. 1005, ¶100; Ex. 1007, ¶26, ¶120; Ex. 1003, ¶193.

The software performs operations taught by Ground 2, including those described below for [13b]-[13g]. The combination further teaches forming "a computer program product" using the software media. Ex. 1005, ¶99; Ex. 1007, ¶119. Therefore, Ground 2 teaches a computer program product as recited in [13a] that performs the operations described below for [13b]-[13g]. Ex. 1003, ¶194.

Limitations [1b]/[7b]/[13b]

Ground 2 teaches [1b], [7b], and [13b]. Ex. 1003, ¶¶195-203.

Karczewicz-I and Karczewicz-II perform motion estimation and motion compensation. Ex. 1005, ¶4, ¶51, ¶53, Fig. 2; Ex. 1007, ¶4, ¶6, ¶54, ¶56, Fig. 2; Ex. 1003, ¶¶195-203.

Motion estimation "generates motion vectors" that point to reference blocks (predictive video blocks) *for a current block* and "indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units." Ex. 1005, ¶7, ¶¶53-54; Ex. 1007, ¶4, ¶56; Ex. 1003, ¶197.

For motion compensation, the encoder determines reference blocks for further calculations and ascertains values of reference blocks, thereby *determining reference blocks* (e.g., predictive video blocks), *based on motion vectors*. Ex. 1005, ¶7, ¶53; Ex. 1007, ¶73, ¶4, ¶56, ¶58; Ex. 1003, ¶198.

Karczewicz-I teaches video encoding using bi-directional prediction "from two lists of data[.]" Ex. 1005, ¶6, ¶5, ¶22, ¶42, ¶54; Ex. 1003, ¶199. The reference data from the two lists include data in two reference blocks and when the current block is a B block having two motion vectors, two reference blocks are determined based on these two motion vectors. Ex. 1005, ¶7, ¶¶53-54. Therefore, the combination teaches *determining, for a current block, a first reference block based* *on a first motion vector and second reference block based on a second motion vector*. Ex. 1003, ¶200.

Karczewicz-II teaches *that the pixels of the current block, the first reference block, and the second reference block have values with a first precision* (8 bits). Ex. 1003, ¶201. "Precision" is at least satisfied by "a number of bits needed to represent possible values." *Supra* §IV.B. Karczewicz-II teaches this precision (Ex. 1007, ¶89; *supra* §IV.B), with 8 bits needed to represent possible pixel values for the two reference blocks. Table 5 shows that integer pixels (*e.g.,* integer pixel x) include possible values between 0-255, and 8-bit unsigned numbers ("8u") are needed to represent these possible values. Ex. 1007, ¶103, Table 5:

TABLE 5

positions {a, c, d, l} of FIGS. 4A-4D				
Operation	Comment	Min value	Max value	Register size
r1 = x	r1 is integer pixel x	0	255	8u
r1 = r1 << 5	r1 is 32 * x	0	8160	13u
r2 = y0	r2 is y0 (y0 is a one-dimensional (1-D) half-pixel such as b1, h1, ee1 and hh1 before shifting down)	−2550	10710	15s
r1 = r1 + r2	r1 is 32 * x + y0	−2550	18870	16s
r1 = r1 + 32	r1 is 32 * x + y0 + 32	−2518	18902	16s
r1 = r1 >> 6	r1 is (32 * x + y0 + 32) >> 6	−39	295	11s
r1 = max (0, r1)	clip r1 on the low side	0	295	10u
r1 = min (255, r1)	clip r1 on the high side	0	255	8u

Ex. 1003, ¶¶201-202.

Therefore, the combination teaches or suggests that the pixels of the current block, the first reference block, and the second reference block have values with a first precision (8 bits). Ex. 1003, ¶203.

Limitations [1c]/[7c]/[13c] and [1d]/[7d]/[13d]

Ground 2 teaches limitations [1c], [7c], and [13c] as well as limitations [1d], [7d], and [13d]. Ex. 1003, ¶¶204-225.

Karczewicz-I teaches calculating a bi-directional prediction using a default weighted prediction mode, with equal weights assigned to two reference blocks

(Ex. 1005, ¶55, ¶24, ¶44, ¶48), calculated as an average of two predictions (pred0(i,j) and pred1(i,j)). Ex. 1005, ¶60:

$$\text{pred}(i,j) = (\text{pred0}(i,j) + \text{pred1}(i,j) + 1) >> 1$$

Ex. 1003, ¶¶205-206.

Here, pred0(i,j) is a first prediction based on a "motion compensated reference area[] … obtained from list 0 … reference picture." Ex. 1005, ¶¶57-58. Similarly, pred1(i,j) is obtained from the list 1 reference picture. *Id.* The motion compensated reference area refers to a reference block since Karczewicz-I teaches bi-directional prediction based on reference blocks. *Supra* §V.B.4.a[1b]/[7b]/[13b]; Ex. 1005, ¶7, ¶¶53-54. The combination of Karczewicz-I and Karczewicz-II does not change the reference blocks on which Karczewicz-I's predictions are based. *Supra* §V.B.3 (explaining combination). Thus, the combination teaches *using said first and second reference blocks* (block from list 0 and list 1 reference pictures) *to obtain first and second predictions*. Ex. 1003, ¶¶208-225.

A POSITA would have understood that, depending on the motion vectors, each prediction would consist either of integer pixels or interpolated subpixels (half- or quarter-pixels). Ex. 1003, ¶208. As discussed, Karczewicz-II teaches using higher-precision intermediate values to generate more accurate predictions and it would have been obvious to a POSITA to use the same approach when averaging pixels to generate the bidirectional weighted prediction. *Supra* §V.B.3.

As one example, a given B-video block might be interpolated from a block of interpolated half-pixels based on the list 0 picture and a block of integer pixels from the list 1 picture. In this scenario (Scenario 1 discussed in §V.B.3), the improved calculation of the default weighted prediction would mirror Karczewicz-II's calculations for averaging half-pixels with integer pixels:

TABLE 5

positions {a, c, d, l} of FIGS. 4A-4D

Operation	Comment	Min value	Max value	Register size
r1 = x	r1 is integer pixel x	0	255	8u
r1 = r1 << 5	r1 is 32 * x	0	8160	13u
r2 = y0	r2 is y0 (y0 is a one-dimensional (1-D) half-pixel such as b1, h1, ee1 and hh1 before shifting down)	−2550	10710	15s
r1 = r1 + r2	r1 is 32 * x + y0	−2550	18870	16s
r1 = r1 + 32	r1 is 32 * x + y0 + 32	−2518	18902	16s
r1 = r1 >> 6	r1 is (32 * x + y0 + 32) >> 6	−39	295	11s
r1 = max (0, r1)	clip r1 on the low side	0	295	10u
r1 = min (255, r1)	clip r1 on the high side	0	255	8u

Ex. 1007, ¶103.

The first prediction (*i.e.,* pred0(*i,j*)), like r2 in the Table above, would consist of (non-rounded) half-pixels. These values also have a higher precision of 13 bits even though they were themselves interpolated using 8-bit integer pixel values.[7] Ex. 1003, ¶212. The second prediction, like r1 in the Table above, would consist of integer pixel values left shifted by five bits. That second prediction would have a precision of 13 bits which is higher than the original 8 bits per integer pixel in the reference image. *Id.*

The improved bidirectional weighted prediction equation for this scenario (Scenario 1) would be:

$$\text{pred}(i,j) = (\text{non-rounded}(\text{pred0}(i,j)) + \text{pred1}(i,j) \ll 5 + 32) \gg 6.$$

Ex. 1003, ¶210.

Limitations [1e]/[7e]/[13e]

Ground 2 teaches [1e], [7e], and [13e]. Ex. 1003, ¶¶226-232.

Karczewicz-I teaches obtaining a combined prediction by averaging the first and second predictions. Ex. 1005, ¶60 (adding predictions and dividing by two with ">>1"):

$$\text{pred}(i,j) = (\text{pred0}(i,j) + \text{pred1}(i,j) + 1) \gg 1$$

Supra §V.B.4.a[1c-1d]/[7c-7d]/[13c-13d]. Ground 2 combines these bi-prediction teachings with Karczewicz-II. *Supra* §V.B.3 (explaining combination); Ex. 1003, ¶¶226-232.

[7] The '267 patent likewise describes obtaining a higher precision prediction by interpolating sub-pixels from integer pixels. Ex. 1001, 12:56-60.

As discussed, it would have been obvious to improve Karczewicz-I's bidirectional default weighted prediction calculation to use higher-precision prediction values consistent with Karczewicz-II's improved pixel-averaging calculations. *Supra* §V.B.3. The improved calculations (*see* table in §V.B.3) sum the higher-precision pixel values to obtain a combined prediction. Ex. 1003, ¶232.

Limitations [1f]/[7f]/[13f]

Ground 2 teaches [1f], [7f], and [13f]. Ex. 1003, ¶¶233-240.

Karczewicz-II's higher-precision interpolation equations increase the extent of the right shift as part of the process for reducing the precision of the combined prediction back to 8 bits. For example, Table 5 reduces the precision of the combined prediction when it applies a right-shift of six bits to the value r1:

TABLE 5

	positions {a, c, d, l} of FIGS. 4A-4D			
Operation	Comment	Min value	Max value	Register size
r1 = x	r1 is integer pixel x	0	255	8u
r1 = r1 << 5	r1 is 32 * x	0	8160	13u
r2 = y0	r2 is y0 (y0 is a one-dimensional (1-D) half-pixel such as b1, h1, ee1 and hh1 before shifting down)	−2550	10710	15s
r1 = r1 + r2	r1 is 32 * x + y0	−2550	18870	16s
r1 = r1 + 32	r1 is 32 * x + y0 + 32	−2518	18902	16s
r1 = r1 >> 6	r1 is (32 * x + y0 + 32) >> 6	−39	295	11s
r1 = max (0, r1)	clip r1 on the low side	0	295	10u
r1 = min (255, r1)	clip r1 on the high side	0	255	8u

Ex. 1005, ¶103; *see also id.*, ¶¶98, 101, 103, 105 (Tables 3, 4, 6 and 8 showing similar right-shifts applied to combined predictions). Ex. 1003, ¶236.

As discussed, it would have been obvious to apply Karczewicz-II's higher-precision pixel-averaging improvements (including the increased right shift of the combined prediction) to Karczewicz-I's bidirectional default weighted prediction calculation. *Supra* § V.B.3. Ex. 1003, ¶235.

Limitations [1g]/[7g]/[13g]

Ground 2 teaches [1g], [7g], and [13g]. Ex. 1003, ¶¶241-243.

The combination teaches determining the combined prediction in the motion compensation process. *Supra* §V.B.4.a[1b-1f]/[7b-7f]/[13b-13f]. The Karczewicz references teach *determining residual data* (e.g., a residual video block) *based upon a difference between the combined prediction* (e.g., prediction data) *and the block of pixels* (e.g., original video block). Ex. 1005, ¶73 ("Video encoder 50 forms a residual video block by subtracting the prediction data from the original video block being coded."), ¶7, ¶89; Ex. 1007, ¶4, ¶6, ¶35, ¶50, ¶58, ¶60, ¶73, Fig. 2. Ex. 1003, ¶242. The combination further teaches *encoding* (e.g., transforming, quantizing, and entropy coding) *residual data in a bitstream*. Ex. 1005, ¶¶73-75, Fig. 2; Ex. 1007, ¶5, ¶35, ¶50, ¶¶60-61, ¶110, Fig. 2; Ex. 1003, ¶243.

b. Dependent Claims 2, 8, and 14

Ground 2 teaches claims 2, 8, and 14. Ex. 1003, ¶¶244-252.

Karczewicz-I discloses that "inter-based predictive encoding … includes interpolation" and incorporates-by-reference the AI33 submission which discusses high-precision interpolation. Ex. 1005, ¶¶41, 37; Ex. 1006. A POSITA would have been familiar with subpixel interpolation. *Supra* §I.B. Accordingly, a POSITA would have understood that Karczewicz-I's biprediction equations would use interpolated reference pixels whenever a motion vector points to a subpixel. Ex. 1003, ¶247. Further, Karczewicz-II provides detailed teaches about obtaining predictions by interpolation using pixel values of reference blocks when motion vectors point to subpixels. Ex. 1007, ¶42 ("The interpolation techniques of this disclosure may be performed by any encoding device that supports motion compensated interpolation to sub-pixel resolution."), ¶7, ¶66, ¶¶68-72, Figs. 4A-4D; Ex. 1003, ¶248.

c. Dependent Claims 3, 9, and 15

(i) Ground 2

Ground 2 teaches claims 3, 9, and 15. Ex. 1003, ¶¶253-258.

Karczewicz-II teaches that "pixel location 'j' is a special case that may require rounding of intermediate results for purposes of generating its half-pixel value since pixel location 'j' requires two levels of half-pixel interpolation." Ex. 1007, ¶106. The intermediate value j1 is calculated using a P-tap filter[8]:

$$\mathrm{j1}=aa1-5*bb1+20*b1+20*hh1-5*ii1+jj1$$

Id., ¶95. Because "twenty bits may be required" for j1, its value is right-shifted 5 bits before averaging it with intermediate values for other half-pixels which are 15 bits (*e.g.,* b1):

$$\begin{aligned} f &= (j1 >> 5 + b1 + 32) >> 6 \\ i &= (j1 >> 5 + h1 + 32) >> 6 \\ k &= (j1 >> 5 + ee1 + 32) >> 6 \\ n &= (j1 >> 5 + hh1 + 32) >> 6 \end{aligned}$$

[8] The intermediate values (*e.g.,* aa1) used to calculate j1 are themselves derived from integer pixel values using a P-tap filter. Ex. 1006, ¶¶93, 95. Ex. 1003, ¶255.

Id., ¶99; *see also id.*, ¶103, ¶104 (Table 7). Moreover, j1 is interpolated from the integer values from the reference picture. *Id.*, ¶95. Ex. 1003, ¶255.

Accordingly, when the first motion vector points to center-pixel 'j' and the second motion vector points to another half-pixel (Scenario 2 in §V.B.3), the first prediction would be obtained by right-shifting a sum of a P-tap filter using values of the first reference block. Ex. 1003, ¶256.

(ii) Ground 2A

Ground 2A which adds Srinivasan also renders obvious claims 3, 9, and 15. Ex. 1003, ¶¶259-262.

As discussed for Ground 1A, Srinivasan describes two-dimensional sub-pixel interpolation in VC-1 which involves partially rounding/right-shifting intermediate results obtained using a P-tap filter applied to integer pixel values from the reference picture. *Supra* §V.A.c.ii.

Karczewicz-II discloses using a variety of different interpolation filters. Ex. 1007, ¶9 ("[D]ifferent numbers of filter coefficients (different number of taps) or different types of filters altogether may be pre-defined, and then selected and used during the encoding and decoding processes."). It would have been obvious to a POSA for Karczewicz-II encoder and decoder to use the VC-1 interpolation filter described by Srinivasan as one of the available interpolation filters. Ex. 1003, ¶260. Accordingly, the two-stage filtering process described by Srinivasan would be used during interpolation of the ½ pixel in location j, for example. The first stage would use the four-tap filter to calculate intermediate values from integer pixel values in the reference frame and then right-shift those intermediate values to limit them to 16 bits. Ex. 1008, 000015; Ex. 1003, ¶260.

A POSITA would have been motivated to combine the prior art in this manner. Both Karczewicz-II and Srinivasan teach the use of higher precision intermediate values during sub-pixel interpolation to improve accuracy. *Compare* Ex. 1007, ¶102 ("By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate.") *with* Ex. 1008, 000015 ("possible to achieve additional accuracy by retaining a higher precision result after the first stage of filtering"). Moreover, both Karczewicz-II and Srinivasan recognize that some rounding may be necessary to limit intermediate values to 16 bits. *Compare* Ex. 1007, ¶59 ("slight rounding may be applied … to ensure that fixed size storage elements (*e.g.,* 16-bit registers) can be used to store any intermediate values") *with* Ex. 1008, 000015 ("shifts are chosen … to allow for a 16 bit implementation — where the intermediate values in the second filtering operation are within 16 bits").

Additionally, a POSITA would have had a reasonable expectation of success in combining these teachings as proposed. Ex. 1003, ¶262. Karczewicz-II discloses partial rounding of intermediate values to fit within 16-bit registers and for averaging with lower-precision intermediate values. Srinivasan discloses a similar approach for the same reasons during sub-pixel interpolation in two dimensions. Karczewicz-II also discloses that a variety of different interpolation filters can be used. Although Srinivasan describes other aspects of VC-1, Petitioners do not propose to incorporate any aspect of VC-1 beyond Srinivasan's teachings regarding high-precision sub-pixel interpolation.

d. Dependent Claims 4, 10, and 16

Ground 2 teaches claims 4, 10, and 16. Ex. 1003, ¶¶263-269.

Karczewicz-II teaches high-precision calculations for averaging integer pixels with interpolated sub-pixels:

TABLE 3

$$
\begin{aligned}
a &= (C3 \ll 5 + b1 + 32) \gg 6 \\
c &= (C4 \ll 5 + b1 + 32) \gg 6 \\
d &= (C3 \ll 5 + h1 + 32) \gg 6 \\
l &= (D3 \ll 5 + h1 + 32) \gg 6
\end{aligned}
$$

Ex. 1007, ¶¶99. In these calculations, the integer pixel values (e.g., C3, C4) are left-shifted by 5 bits to match the increased precision of the interpolated sub-pixels (*e.g.*, b1, h1). Ex. 1003, ¶266.

As discussed, it would have been obvious to apply Karczewicz-II's improvements to Karczewicz-I's calculations which similarly take an average of two pixels but for the purpose of biprediction. *Supra* §V.B.3. It would have been obvious to apply a left shift to the integer pixel values when the second motion vector for the list 1 reference image points to an integer pixel and the first motion vector for the list 0 reference image points to an interpolated half-pixel (Scenario 1 in §V.B.3). Ex. 1003, ¶268.

e. Dependent Claims 5, 11, and 17

Ground 2 teaches claims 5, 11, and 17. Ex. 1003, ¶¶270-275.

Karczewicz-I's bidirectional default weighted prediction calculation adds a rounding offset before applying the right-shift to the sum of the reference pixels.

$$\text{pred}(i,j) = (\text{pred0}(i,j) + \text{pred1}(i,j) + 1\ ) >> 1$$

Ex. 1005, ¶60; ¶63 (noting that "a rounding adjustment of 1 is used in the default weighted prediction").

Karczewicz-II's high-precision calculations use a larger rounding adjustment because of the increased precision of the pixel values:

TABLE 3

$$a = (C3 << 5 + b1 + 32) >> 6$$
$$c = (C4 << 5 + b1 + 32) >> 6$$
$$d = (C3 << 5 + h1 + 32) >> 6$$
$$l = (D3 << 5 + h1 + 32) >> 6$$

Ex. 1007, ¶99; *see also* Ex. 1005, ¶63 ("a rounding adjustment of 2^{r-1} is commonly used prior to a right shift by *r*"). Ex. 1003, ¶135.

As discussed, it would have been obvious to apply Karczewicz-II's improvements to Karczewicz-I's calculations. *Supra* §V.A.3. In doing so, consistent with both references, a rounding offset would be added to the combined prediction before decreasing the precision of the combined prediction using the right-shift. Ex. 1003, ¶273.

Because the combination applies its operations to a computer implementation, it teaches 1) at least one memory and computer program code being configured to cause the apparatus to perform the operations of claim 11 (*supra* §V.B.4[7a]); and 2) program code instructions configured to perform the operations of claim 17 (*supra* §V.B.4.a[13a]). Ex. 1003, ¶¶274-275.

f. Dependent Claims 6, 12, and 18

Ground 2 teaches claims 6, 12, and 18. Ex. 1003, ¶¶276-278.

First, the first precision indicates a number of bits needed to represent the values of the pixels. The combination teaches that the pixels of the current block, the first reference block, and the second reference block have values with a first precision because 8 bits are needed to represent the possible pixel values of these blocks. *Supra* §V.B.4.a[1b]/[7b]/[13b]; Ex. 1003, ¶277.

Second, the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction. The combination teaches said first prediction and second prediction having a second precision that is higher than said first precision because more bits are needed to represent the possible values of the predictions under each of Scenarios 1, 2, and 3. *Supra* §V.B.4.a[1c&d]/[7c&d]/[13c&d]; Ex. 1003, ¶278.

5. Decoder Claims

The decoder-side claims of the ’267 patent recite almost identical limitations to the encoder-side claims discussed above. Moreover, the prior art teaches that bidirectional prediction and interpolation is performed during decoding the same way as during encoding. Karczewicz-I’s decoder performs “the reciprocal decoding techniques to the encoding techniques described” for the encoder including support for weighted prediction. Ex. 1005, ¶83, *see also id.*, ¶¶84-86, 88-89 (discussing decoding including block based predictive decoding using weighted prediction data); ¶44 (video decoder 28 supports “different types of weighted bi-directional prediction in accordance with ITU-T H.264”). Karczewicz-II similarly discloses the use of the same higher precision non-rounded interpolated pixel values for quarter-pixel interpolation *during decoding*. Ex. 1007, ¶¶53, 59, 113 (Fig. 7 process performed during encoding and decoding); Fig. 7.

Accordingly, where appropriate, the discussion below for the decoder-side claims will refer back to the detailed discussion above of similar limitations of the encoder-side claims.

a. Independent Claims 19, 25, and 31

Preamble [19a]

Ground 2 teaches [19a]. Ex. 1003, ¶¶281-284.

Karczewicz-I "describes video encoding and decoding techniques applicable to bi-directional prediction" including for encoders and decoders. Ex. 1005, ¶8, ¶22, ¶¶29-30, ¶33, Fig. 1. The decoder "perform[s] the reciprocal decoding techniques to the encoding techniques."[9] Ex. 1005, ¶¶83-86, Fig. 4; Ex. 1003, ¶282. Karczewicz-I decodes "blocks of pixel data[.]" Ex. 1005, ¶¶39-40; Ex. 1003, ¶282.

Karczewicz-II "describes various interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding." Ex. 1007, Abstract, ¶2, ¶6, ¶¶40-41, 44, ¶¶63-66, Figs. 1, 3. Karczewicz-II decodes a block of pixels. Ex. 1007, ¶64, ¶66, ¶50; Ex. 1003, ¶283.

It would have been obvious to modify Karczewicz-I's decoding method based on Karczewicz-II's interpolation techniques. *Supra* §V.B.3 (explaining combination). The combination teaches a method for decoding a block of pixels, comprising the operations explained below for [19b]-[19g]. Ex. 1003, ¶284.

Preamble [25a]

Ground 2 teaches [25a]. Ex. 1003, ¶¶285-290.

Ground 2 teaches a video decoder for decoding a block of pixels that performs operations described below for [25b]-[25g]. *Supra* §V.B.4[19a]

[9] Karczewicz-I explains that its teachings regarding bi-directional prediction (e.g., ¶¶55-60) are applicable to video decoding. Ex. 1005, ¶8, ¶35.

(explaining decoder); Ex. 1005, ¶¶29-30, ¶33, Fig. 1, ¶¶83-86, Fig. 4; Ex. 1007, ¶¶15-17, ¶¶22-¶23, ¶¶40-41, ¶44, Fig. ¶¶63-66, Fig. 3:

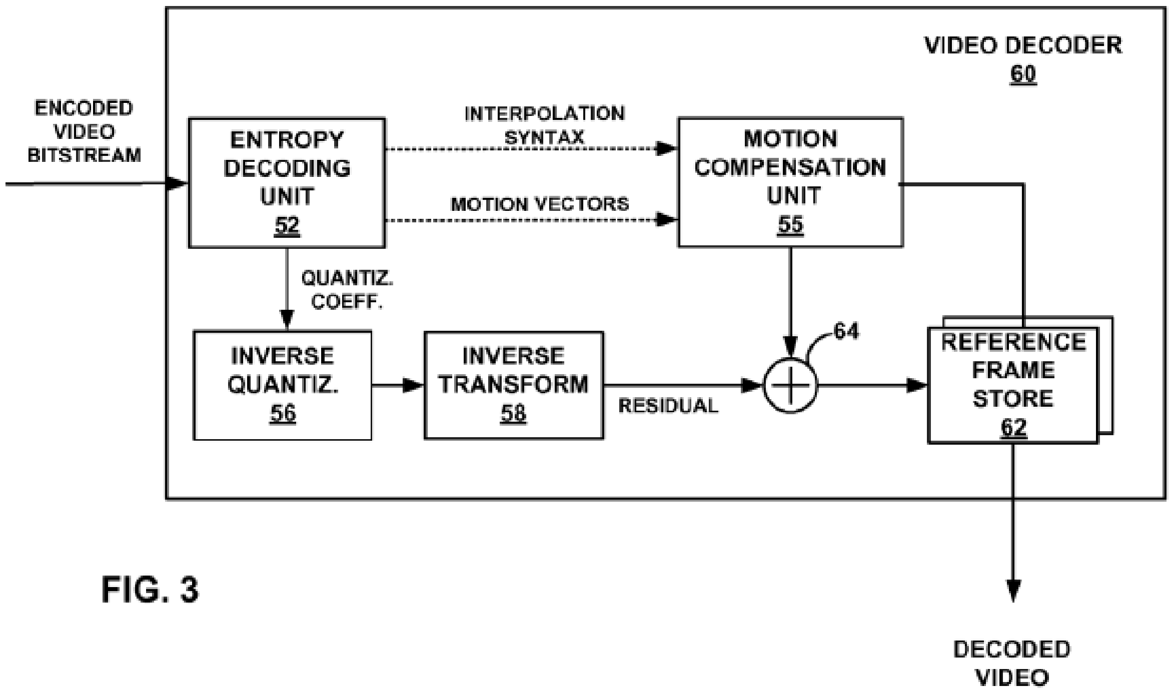


FIG. 3

Ex. 1003, ¶285.

The video decoder decodes a block of pixels. Ex. 1005, ¶¶39-40; Ex. 1007, ¶50, ¶64, ¶66. Karczewicz-I and Karczewicz-II implement their decoder teachings in various types of devices (Ex. 1005, ¶3; Ex. 1007, ¶3) using a processor and memory (computer readable medium) that includes software, which comprises computer program code. Ex. 1005, ¶12, ¶38, ¶¶98-100; Ex. 1007, ¶¶25-27, ¶¶118-120. The medium comprises well-known memory types (e.g., RAM, ROM). Ex. 1005, ¶99; Ex. 1007, ¶119; Ex. 1003, ¶¶287-288.

Karczewicz-I and Karczewicz-II teach the processor executing computer program code in memory to carry out their teachings. Ex. 1005, ¶¶99-100; Ex. 1007, ¶48, ¶¶119-120. The apparatus further performs operations described below for [25b]-[25g]. Ex. 1003, ¶¶289-290.

Preamble [31a]

Ground 2 teaches [31a]. Ex. 1003, ¶¶291-293.

As explained above, Karczewicz-I and Karczewicz-II implement their video-decoder teachings and perform operations for decoding blocks of pixels using software stored in computer-readable storage media and executed by a processor. *Supra* §V.B.4[25a]; Ex. 1005, ¶12, ¶38, ¶¶98-100; Ex. 1007, ¶¶25-26, ¶48, ¶¶119-120. These types of media (e.g., RAM, ROM) are non-transitory. Ex. 1005, ¶99; Ex. 1007, ¶119. The software includes computer-executable program code portions that comprise program code instructions. Ex. 1005, ¶100; Ex. 1007, ¶26, ¶120; Ex. 1003, ¶292.

The software performs operations taught by Ground 2, including those described below for [31b]-[31g]. The combination further teaches forming "a computer program product" using the software media. Ex. 1005, ¶99; Ex. 1007, ¶119. Therefore, Ground 2 teaches a computer program product as recited in [31a] that performs the operations described below for [31b]-[31g]. Ex. 1003, ¶293.

Limitations [19b]/[25b]/[31b]

Limitations [19b], [25b], and [31b] are identical to limitations [1b], [7b], and [13b]. Ground 2 teaches or suggests these limitations as discussed. Supra §V.B.4.a. Ex. 1003, ¶294.

Limitations [19c]/[25c]/[31c]

Limitations [19c], [25c], and [31c] are identical to limitations [1c], [7c], and [13c]. Ground 2 teaches or suggests these limitations as discussed. Supra §V.B.4.a. Ex. 1003, ¶295.

Limitations [19d]/[25d]/[31d]

Limitations [19d], [25d], and [31d] are identical to limitations [1d], [7d], and [13d]. Ground 2 teaches or suggests these limitations as discussed. Supra §V.B.4.a. Ex. 1003, ¶295.

Limitations [19e]/[25e]/[31e]

Limitations [19e], [25e], and [31e] are identical to limitations [1e], [7e], and [13e]. Ground 2 teaches or suggests these limitations as discussed. Supra §V.B.4.a. Ex. 1003, ¶296.

Limitations [19f]/[25f]/[31f]

Limitations [19f], [25f], and [31f] are identical to limitations [1f], [7f], and [13f]. Ground 2 teaches or suggests these limitations as discussed. *Supra* §V.B.4.a. Ex. 1003, ¶297.

Limitations [19g]/[25g]/[31g]

Ground 2 teaches [19g], [25g], and [31g]. Ex. 1003, ¶¶298-299.

The combination teaches determining the combined prediction in the motion compensation process. *Supra* §V.B.4[19b-19f]/[25b-25f]/[31b-31f]. Karczewicz-I teaches **reconstructing the block of pixels** (reconstructed video block) **based on the combined prediction** (prediction block). Ex. 1005, ¶86 ("Adder 79 combines the prediction data (e.g., a prediction block) generated by prediction unit 75 with the residual block from inverse transform unit 78 to create a reconstructed video block[.]"), ¶89; Figs. 4, 6. Karczewicz-II includes similar teachings. Ex. 1007, ¶6, ¶65, ¶111, Figs. 3, 6; Ex. 1003, ¶299.

b. Dependent Claims 20, 26, and 32

These claims add the same limitations as encoder-side dependent claims 2, 8, and 14 and are obvious for the same reasons. *Supra* §V.B.4.b. Ex. 1003, ¶300.

c. Dependent Claims 21, 27, and 33

These claims add the same limitations as encoder-side dependent claims 3, 9, and 15 and Grounds 2 and 2A render them obvious for the same reasons. *Supra* §V.B.4.c. Ex. 1003, ¶301.

d. Dependent Claims 22, 28, and 34

These claims add the same limitations as encoder-side dependent claims 4, 10, and 16 and are obvious for the same reasons. *Supra* §V.B.4.d. Ex. 1003, ¶302.

e. Dependent Claims 23, 29, and 35

These claims add the same limitations as encoder-side dependent claims 5, 11, and 17 and are obvious for the same reasons. *Supra* §V.B.4.e. Ex. 1003, ¶303.

f. Dependent Claims 24, 30, and 36

These claims add the same limitations as encoder-side dependent claims 6, 12, and 18 and are obvious for the same reasons. *Supra* §V.B.4.f. Ex. 1003, ¶304.

VI. MANDATORY NOTICES AND FEES

A. Real Party-In-Interest:

Petitioners identify themselves as real parties in interest. Out of an abundance of caution, Petitioner Hisense USA Corporation also identifies Hisense Co., Ltd., which entity was also sued by Nokia for alleged infringement of the ’267 patent.

B. Related Matters:

To the best of Petitioners’ knowledge, the ’267 patent is presently or has been involved in the following related matters:

Case	Filing Date
Certain Video-Capable Laptop, Desktop Computers, Handheld Computers, Tablets, Televisions, Projectors, and Components and Modules Thereof Inv. No. 337-TA-1448 (ITC)	4/11/2025
Nokia Technologies Oy v. Hisense Co. Ltd. et al No. 1-25-cv-01871 (NDGA)	4/7/2025
Nokia Technologies Oy v. ASUSTeK Computer Inc. et al No. 2-25-cv-03053 (CDCA)	4/7/2025

Nokia Technologies Oy v. Acer Inc. et al No. 1-25-cv-00523 (WDTX)	4/7/2025
Element Television Company, LLC et al v. Nokia Corporation a/k/a Nokia of America Corporation et al No. 0-24-cv-04269 (DMN)	11/25/2024
Certain Video Capable Electronic Devices, Including Computers, Streaming Devices, Televisions, and Components and Modules Thereof Inv. No. 337-TA-1380 (ITC)	10/31/2023
Nokia Corporation et al v. Amazon.com, Inc. No. 1-23-cv-01232 (DDE)	10/27/2023
Nokia Technologies Oy v. HP Inc. f/k/a Hewlett-Packard Company No. 1-23-cv-01237 (DDE)	10/31/2023
Nokia Technologies Oy v. Amazon.com, Inc. et al No. 1-23-cv-01236 (DDE)	10/31/2023
ASUSTeK Computer Inc. et al. v. Nokia Corporation and Nokia Technologies Oy, IPR2025-01154 (PTAB)	6/17/2025
Amazon.com, Inc. et al. v. Nokia Technologies Oy, IPR2024-00626 (PTAB)	3/15/2024
Amazon.com, Inc. et al. v. Nokia Technologies Oy, IPR2024-00627 (PTAB)	3/15/2024

C. Notice of Counsel and Service Information

LEAD COUNSEL
Patrick J. McKeever (Reg. No. 66,019) PERKINS COIE LLP 11452 El Camino Real, Suite 300 San Diego, CA 92130-2080 Telephone: (858) 720-5722 mckeever-ptab@perkinscoie.com Hisense-NokiaIPR@perkinscoie.com

BACK-UP COUNSEL	
Brian E. Ferguson (Reg. No. 36,801) WINSTON & STRAWN LLP 1901 L Street, N.W. Washington, D.C. 20036 T: 202-282-5276 beferguson@winston.com Joseph C. Masullo (*pro hac vice to be submitted*) WINSTON & STRAWN LLP 1901 L Street, N.W. Washington, D.C. 20036 T: 202-282-5897 jmasullo@winston.com Eimeric Reig (*pro hac vice to be submitted*) WINSTON & STRAWN LLP 101 California Street 21st Floor San Francisco, CA 94111 T: 415-591-6808 ereigplessis@winston.com Lauren M. Rennecker (*pro hac vice to be submitted*) WINSTON & STRAWN LLP 1901 L Street, N.W. Washington, D.C. 20036 T: 202-282-5897 lrennecker@winston.com Attorneys for Petitioner Snap Inc.	Wei Yuan (Reg. No. 71,772) PERKINS COIE LLP 11452 El Camino Real, Suite 300 San Diego, CA 92130-2080 Telephone: (858) 720-5798 yuan-ptab@perkinscoie.com Nicholas Lipperd (Reg. No. 83,084) PERKINS COIE LLP 11452 El Camino Real, Suite 300 San Diego, CA 92130-2080 Telephone: (858) 720-5798 lipperd-ptab@perkinscoie.com Attorneys for Petitioner Hisense USA Corporation

Respectfully submitted,

PERKINS COIE LLP

/ Patrick J. McKeever /

Patrick J. McKeever
Reg. No. 66,019

PERKINS COIE LLP
11452 El Camino Real, Suite 300
San Diego, CA 92130-2080

CLAIM APPENDIX

[1a] A method for encoding a block of pixels, the method comprising:

[1b] determining, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

[1c] using said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[1d] using said second reference block to obtain a second prediction, said second prediction having the second precision;

[1e] obtaining a combined prediction based at least partly upon said first prediction and said second prediction;

[1f] decreasing a precision of said combined prediction by shifting bits of the combined prediction to the right; and

[1g] encoding residual data in a bitstream, wherein the residual data is determined based upon a difference between the combined prediction and the block of pixels.

2. The method according to claim 1, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.

3. The method according to claim 2, wherein said first prediction is obtained by interpolation using values of said first reference block by:

right shifting a sum of a P-tap filter using values of said first reference block.

4. The method according to claim 2, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

5. The method according to claim 1, wherein said decreasing said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprises: inserting a rounding offset to the combined prediction before said decreasing.

6. The method according to claim 1, wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

[7a] An apparatus for encoding a block of pixels, the apparatus comprising: at least one processor and at least one memory including computer program code, the at least one memory and computer program code configured to, with the at least one processor, cause the apparatus to:

[7b] determine, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector,

wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

[7c] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[7d] use said second reference block to obtain a second prediction, said second prediction having the second precision;

[7e] obtain a combined prediction based at least partly upon said first prediction and said second prediction;

[7f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and

[7g] encode residual data in a bitstream, wherein the residual data is determined based upon a difference between the combined prediction and the block of pixels.

8. The apparatus according to claim 7, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.

9. The apparatus according to claim 8, wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.

10. The apparatus according to claim 8, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

11. The apparatus according to claim 7, wherein the at least one memory and computer code are configured to cause the apparatus to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, by: inserting a rounding offset to the combined prediction before said decreasing.

12. The apparatus according to claim 7, wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

[13a] A computer program product for encoding a block of pixels, the

computer program product comprising at least one non-transitory computer readable storage medium having computer executable program code portions stored therein, the computer executable program code portions comprising program code instructions configured to:

[13b] determine, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

[13c] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[13d] use said second reference block to obtain a second prediction, said second prediction having the second precision;

[13e] obtain a combined prediction based at least partly upon said first prediction and said second prediction;

[13f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and

[13g] encode residual data in a bitstream, wherein the residual data is determined based upon a difference between the combined prediction and the block of pixels.

14. The computer program product according to claim 13, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.

15. The computer program product according to claim 14, wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.

16. The computer program product according to claim 14, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

17. The computer program product according to claim 13, wherein the program code instructions configured to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprise program code instructions configured to: insert a rounding offset to the combined prediction before said decreasing.

18. The computer program product according to claim 13, wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

[19a] A method for decoding a block of pixels, the method comprising:

[19b] determining, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

[19c] using said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[19d] using said second reference block to obtain a second prediction, said second prediction having the second precision;

[19e] obtaining a combined prediction based at least partly upon said first prediction and said second prediction;

[19f] decreasing a precision of said combined prediction by shifting bits of the combined prediction to the right; and

[19g] reconstructing the block of pixels based on the combined prediction.

20. The method according to claim 19, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.

21. The method according to claim 20, wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.

22. The method according to claim 20, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

23. The method according to claim 19, wherein said decreasing said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprises: inserting a rounding offset to the combined prediction before said decreasing.

24. The method according to claim 19, wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

[25a] An apparatus for decoding a block of pixels, the apparatus comprising:

at least one processor and at least one memory including computer program code, the at least one memory and computer program code configured to, with the at least one processor, cause the apparatus to:

[25b] determine, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

[25c] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[25d] use said second reference block to obtain a second prediction, said second prediction having the second precision;

[25e] obtain a combined prediction based at least partly upon said first prediction and said second prediction;

[25f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and

[25g] reconstruct the block of pixels based on the combined prediction.

26. The apparatus according to claim 25, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.

27. The apparatus according to claim 26, wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.

28. The apparatus according to claim 26, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

29. The apparatus according to claim 25, wherein the at least one memory and computer code are configured to cause the apparatus to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, by: inserting a rounding offset to the combined prediction before said decreasing.

30. The apparatus according to claim 25, wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

[31a] A computer program product for decoding a block of pixels, the computer program product comprising at least one non-transitory computer readable storage medium having computer executable program code portions stored therein, the computer executable program code portions comprising program code instructions configured to:

[31b] determine, for a current block, a first reference block based on a first motion vector and a second reference block based on a second motion vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;

[31c] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision;

[31d] use said second reference block to obtain a second prediction, said second prediction having the second precision;

[31e] obtain a combined prediction based at least partly upon said first prediction and said second prediction;

[31f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and

[31g] reconstruct the block of pixels based on the combined prediction.

32. The computer program product according to claim 31, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.

33. The computer program product according to claim 32, wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.

34. The computer program product according to claim 32, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.

35. The computer program product according to claim 31, wherein the program code instructions configured to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprise program code instructions configured to: insert a rounding offset to the combined prediction before said decreasing.

36. The computer program product according to claim 31, wherein the first precision indicates a number of bits needed to represent the values of the pixels, and the second precision indicates the number of bits needed to represent values of said first prediction and values of said second prediction.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that on October 3, 2025, true and correct copies of the foregoing Petition for Inter Partes Review of U.S. Patent No. 11,805,267 and accompanying documents (Petitioner Powers of Attorney and Exhibits 1001-1028) were served in entirety on the Patent Owner at the following address of record as listed on Patent Center, via Federal Express:

Nokia Corporation and Alston & Bird LLP
Vantage South End
1120 South Tryon Street, Suite 300
Charlotte, NC 28203-6818
Official Correspondence Address

The undersigned certifies that true and correct courtesy copies of the foregoing Petition for Inter Partes Review of U.S. Patent No. 11,805,267 accompanying documents (Petitioner Powers of Attorney and Exhibits 1001-1028) were sent via electronic mail to the attorneys of record for Plaintiff in the concurrent litigation matter(s):

Brian E. Farnan
bfarnan@farnanlaw.com
Michael J. Farnan
mfarnan@farnanlaw.com
John D Haynes
john.haynes@alston.com
Nicholas T. Tsui
nick.tsui@alston.com
Shawn P. Gannon
shawn.gannon@alston.com
Theodore Stevenson, III

Warren H. Lipschitz
wlipschitz@mckoolsmith.com
Alexandra F. Easley
aeasley@mckoolsmith.com
Joshua Budwin
jbudwin@mckoolsmith.com
R. Mitch Verboncoeur
mverboncoeur@mckoolsmith.com
Joshua Newcomer
jnewcomer@mckoolsmith.com
Kevin Burgess

ted.stevenson@alston.com kburgess@mckoolsmith.com
Stephen R. Lareau
stephen.lareau@alston.com

PERKINS COIE

/ Patrick J. McKeever /
Patrick J. McKeever
Reg. No. 66,019

Date: October 3, 2025

11452 El Camino Real Suite 300,
San Diego, CA
Tel: (858) 720-5700

CERTIFICATE OF COMPLIANCE WITH TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS, AND TYPE STYLE REQUIREMENTS

1. This Petition complies with the type-volume limitation of 14,000 words, comprising 13,998 words, as counted using the Microsoft Word software that was used to prepare this paper, excluding the parts exempted by 37 C.F.R. § 42.24(a).

2. This Petition complies with the general format requirements of 37 C.F.R. § 42.6(a) and has been prepared using Microsoft® Word in 14-point Times New Roman.

Respectfully submitted,

PERKINS COIE

/ Patrick J. McKeever /

Patrick J. McKeever
Reg. No. 66,019

Date: October 3, 2025

11452 El Camino Real Suite 300,
San Diego, CA
Tel: (858) 720-5700