# EXHIBIT E-5

# KARCZEWICZ II

# ANTICIPATES / RENDERS OBVIOUS U.S. PATENT NO. 11,805,267

As described in the following claim chart, claims 1-36 of U.S. Patent No. 11,805,267 (the "'267 patent") are invalid in view of US Patent Application Publication 2009/0257499 ("Karczewicz II") (RESPNOKIA_00025222), entitled "Advanced Interpolation Techniques for Motion Compensation in Video Coding," published on October 15, 2009. Karczewicz II was filed April 8, 2009, and is prior art under at least 35 U.S.C. §§ 102(a)/(b)/(e).

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| 1[preamble] A method for encoding a block of pixels, the method comprising: | Karczewicz II, as evidenced by the example citations below, discloses a method for encoding a block of pixels.<br><br>*See, e.g.*,<br><br>Karczewicz II, Abstract:<br><br>> This disclosure describes various interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding. In one example, an encoder interpolates pixel values of reference video data based on a plurality of different pre-defined interpolation filters. In this example, the decoder receives a syntax element that identifies an interpolation filter, and interpolates pixel values of reference video data based on |

To the extent that these Invalidity Contentions rely on or otherwise embody particular constructions of terms or phrases in the Asserted Claims, Respondents are not proposing any such constructions as proper constructions of those terms or phrases. Various positions put forth in this document are predicated on Complainants' incorrect and overly broad interpretation of its claims. Those positions are not intended to and do not necessarily reflect Respondents' interpretation of the true and proper scope of Complainants' claims, and Respondents reserve the right to adopt claim construction positions that differ from or even conflict with various positions put forth in this document.



**Nokia Exhibit 2011, p. 1**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | the interpolation filter identified by the syntax element. In another example, a method of interpolating predictive video data includes generating half-pixel values based on integer pixel values, rounding the half-pixel values to generate half-pixel interpolated values, storing the half-pixel values as non-rounded versions of the half-pixel values, and generating quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0002:<br><br>This disclosure relates to digital video coding and, more particularly, fractional interpolations of predictive data used in video coding.<br><br>Karczewicz II, ¶0004:<br><br>Block based inter-coding is a very useful coding technique that relies on temporal prediction to reduce or remove temporal redundancy between video blocks of successive coded units of a video sequence. The coded units may comprise video frames, slices of video frames, groups of pictures, or another defined unit of video blocks. For inter-coding, the video encoder performs motion estimation and motion compensation to track the movement of corresponding video blocks of two or more adjacent coded units. Motion estimation generates motion vectors, which indicate the displacement of video blocks relative to corresponding prediction video blocks in one or more reference frames or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the reference frame or other coded unit. After motion compensation, residual video blocks are formed by subtracting prediction video blocks from the original video blocks being coded. |



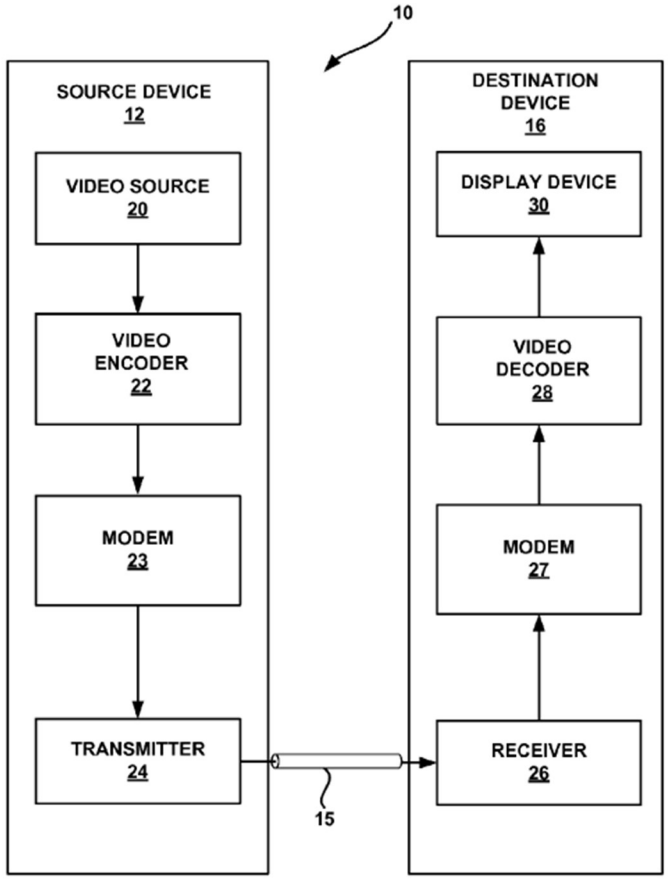
Nokia Exhibit 2011, p. 2
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0012:<br><br>In one example, this disclosure provides a method of encoding video data. The method comprises generating prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on a plurality of different pre-defined interpolation filters. The method also comprises encoding the video data based on the prediction data.<br><br>Karczewicz II, ¶0014:<br><br>In another example, this disclosure provides a method of interpolating predictive video data for video coding. The method comprises generating half-pixel values based on integer pixel values, rounding the half-pixel values to generate half-pixel interpolated values, storing the half-pixel values as non-rounded versions of the half-pixel values, and generating quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0035:<br><br>This disclosure describes various interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding. According to one aspect of this disclosure, the encoder may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data. The interpolated prediction data that achieves the highest levels of compression may be selected at the encoder, and the interpolation filter that was used may be coded as syntax and communicated to the decoding device as part of the coded video data. The interpolated prediction data comprises reference data. Video data to be coded (e.g., a video block) may be subtracted from interpolated prediction data (e.g., an interpolated predictive reference block) in order to define a residual |

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | block of encoded data, which may then be transformed, quantized and entropy coded.<br><br>Karczewicz II, Fig.1: |



Nokia Exhibit 2011, p. 4
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| |  <br><br>FIG. 1 |



Nokia Exhibit 2011, p. 5
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0041:<br><br>In the example of FIG. 1, source device 12 may include a video source 20, video encoder 22, a modulator/demodulator (modem) 23 and a transmitter 24. Destination device 16 may include a receiver 26 a modem 27, a video decoder 28, and a display device 30. In accordance with this disclosure, video encoder 22 of source device 12 may be configured to apply one or more of the interpolation techniques of this disclosure as part of a video encoding process. Similarly, video decoder 28 of destination device 16 may be configured to apply one or more of the interpolation techniques of this disclosure as part of a video decoding process.<br><br>Karczewicz II, Fig. 2: |



Nokia Exhibit 2011, p. 6
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

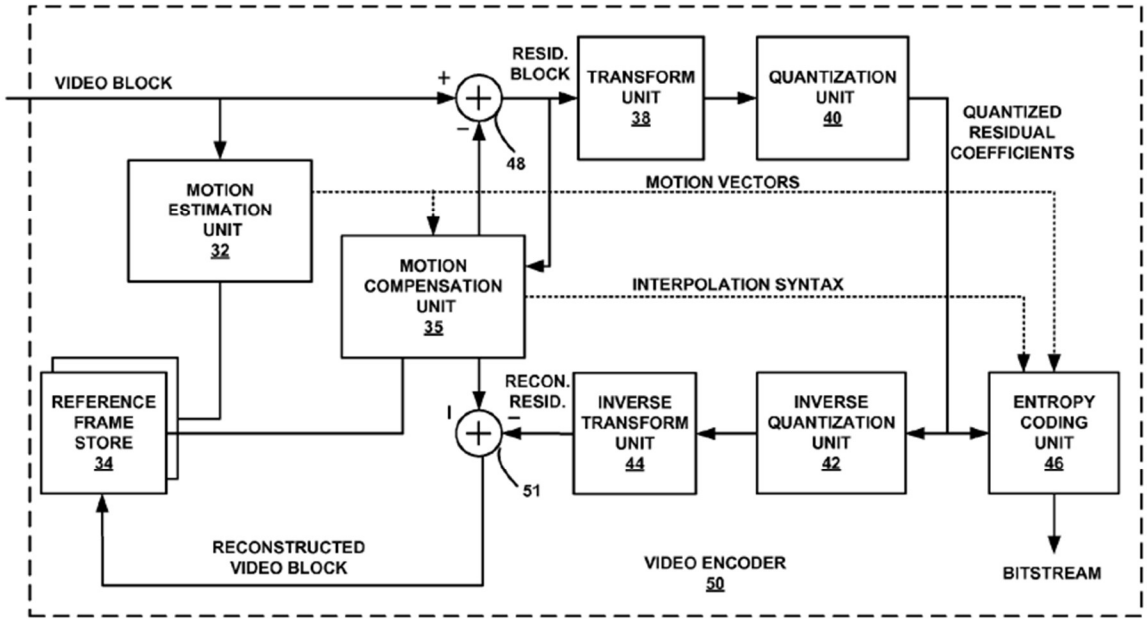



FIG. 2

Karczewicz II, ¶0049:

> A video sequence typically includes a series of video frames. Video encoder 22 operates on video blocks within individual video frames in order to encode the video data. The video blocks may have fixed or varying sizes, and may differ in size according to a specified codingsee standard. Each video frame includes a series of slices. Each slice may include a series of macroblocks, which may be arranged into sub-blocks. As an example, the ITU-T



Nokia Exhibit 2011, p. 7
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | H.264 standard supports intra prediction in various block sizes, such as 16 by 16, 8 by 8, or 4 by 4 for luma components, and 8×8 for chroma components, as well as inter prediction in various block sizes, such as 16 by 16, 16 by 8, 8 by 16, 8 by 8, 8 by 4, 4 by 8 and 4 by 4 for luma components and corresponding scaled sizes for chroma components. Video blocks may comprise blocks of pixel data, or blocks of transformation coefficients, e.g., following a transformation process such as discrete cosine transform or a conceptually similar transformation process.<br><br>Karczewicz II, ¶0050:<br><br>Smaller video blocks can provide better resolution, and may be used for locations of a video frame that include high levels of detail. In general, macroblocks (MBs) and the various sub-blocks may be considered to be video blocks. In addition, a slice may be considered to be a series of video blocks, such as MBs and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Video encoder 22 and video decoder 28 perform inter-based predictive coding, which involves the generation of predictive reference data and the subtraction of a video block to be coded from the predictive reference data to generate residual data, which may then be transformed, quantized and entropy coded. The inter-based predictive coding may include interpolation of the predictive data in accordance with this disclosure.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 1[a] determining, for a current block, a first reference block | Karczewicz II, as evidenced by the example citations below, discloses determining, for a current block, a first reference block based on a first motion vector and a second reference block based on a |



Nokia Exhibit 2011, p. 8
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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; | 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.<br><br>Karczewicz II, ¶0004:<br><br>> Block based inter-coding is a very useful coding technique that relies on temporal prediction to reduce or remove temporal redundancy between video blocks of successive coded units of a video sequence. The coded units may comprise video frames, slices of video frames, groups of pictures, or another defined unit of video blocks. For inter-coding, the video encoder performs motion estimation and motion compensation to track the movement of corresponding video blocks of two or more adjacent coded units. Motion estimation generates motion vectors, which indicate the displacement of video blocks relative to corresponding prediction video blocks in one or more reference frames or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the reference frame or other coded unit. After motion compensation, residual video blocks are formed by subtracting prediction video blocks from the original video blocks being coded.<br><br>Karczewicz II, Fig. 2: |



Nokia Exhibit 2011, p. 9
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

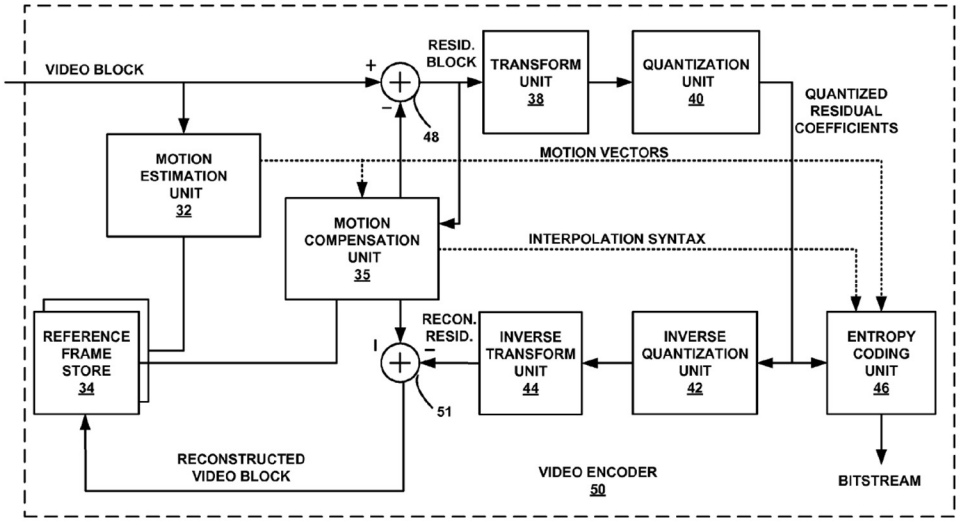



FIG. 2

Karczewicz II, ¶0056:

> During the encoding process, video encoder 50 receives a video block to be coded, and motion estimation unit 32 and motion compensation unit 35 perform inter-predictive coding. Motion estimation unit 32 and motion compensation unit 35 may be highly integrated, but are illustrated separately for conceptual purposes. Motion estimation is typically considered the process of generating motion vectors, which estimate motion for video blocks. A motion vector, for example, may indicate the displacement of a predictive block within a predictive frame (or other coded unit) relative to the current block being coded within the current frame (or other coded unit). Motion compensation is typically considered the process of fetching or

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | generating the predictive block based on the motion vector determined by motion estimation. Again, motion estimation unit 32 and motion compensation unit 35 may be functionally integrated. The interpolation techniques described in this disclosure are described as being performed by motion compensation unit 35. However, interpolation may be performed during motion estimation in order to facilitate the selection of the best motion vectors.<br><br>Karczewicz II, ¶0057:<br><br>In accordance with this disclosure, motion estimation 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of a predictive coded unit (e.g., a previous frame). At this point, motion compensation unit 35 may perform interpolation in order to generate predictive data at sub-pixel resolution. In some cases, during motion estimation, the interpolation may be based on a fixed interpolation filter. In other cases, the different interpolation filters applied during motion compensation (as outlined below) may also be used during the motion estimation process for purposes of motion vector selection.<br><br>Karczewicz II, ¶0058:<br><br>Once motion estimation unit 32 has selected the motion vector for the video block to be coded, motion compensation unit 35 generates the predictive video block associated with that motion vector. According to this disclosure, however, motion compensation unit 35 may consider several versions of any predictive video block that has sub-pixel resolution. In this case, motion compensation unit 35 may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data for the video block to be coded. Motion compensation unit 35 then selects the interpolated prediction data (e.g., an interpolated video block associated with one of the interpolation filters) that achieves the highest levels of compression. The interpolation filter that was used to generate the |



Nokia Exhibit 2011, p. 11
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | interpolated data may be coded as interpolation syntax and communicated to entropy coding unit 46 for inclusion in the coded bitstream. Once motion compensation unit 35 has selected and applied the best interpolation filter, the motion compensation unit 35 generates the predictive data using that interpolation filter, and video encoder applies adder 48 to subtract that predictive data from the video block being coded to generate residual data.<br><br>Karczewicz II, ¶0073:<br><br>After motion estimation, the best motion vector for a given video block may be identified, e.g., possibly using a rate-distortion model to balance the coding rate and quality. Then, the prediction video block is formed during motion compensation using the best motion vector. As outlined above, the residual video block is formed by subtracting the prediction video block from the original video block. A transform is then applied on the residual block, and the transform coefficients are quantized and entropy coded to further reduce bit rate.<br><br>Karczewicz II, ¶0089:<br><br>The average filter for a particular sub-pixel position may also be normalized such that all filter coefficients add up to 1.0. The average filter may also be quantized to a certain fixed-point precision (e.g., 13-bit precision). The use of fixed-point precision in the filter coefficient ensures that implementations across different platforms will not have drifts. Furthermore, a true fixed-point implementation may be derived from such pre-defined filter coefficients. Any customized filters may have different characteristics. Specifically, different filters may be non-separable filters or separable filters, and different filters may define different filter supports (e.g., 6×6 or 4×4). |



Nokia Exhibit 2011, p. 12
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0103:<br><br>Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 13
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |



Nokia Exhibit 2011, p. 14
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 1[b] using said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision; | Karczewicz II, as evidenced by the example citations below, discloses using the first reference block to obtain a first prediction, the first prediction having a second precision, which is higher than the first precision.<br><br>*See, e.g.*,<br><br>Karczewicz II, ¶0010:<br><br>> In addition, this disclosure also recognizes coding inefficiencies due to conventional rounding of half-pixel values, and provides techniques that may improve interpolation by reducing or eliminating intermediate rounding. In this case, interpolated half-pixel values may be rounded for purposes of half-pixel interpolation. Quarter-pixel values, however, which may be generated based on one or more of the interpolated half-pixel values, may rely on non-rounded versions of the half-pixel values. This can eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding without sacrificing the accuracy of the final values may be applied to one specific half-pixel value in order to ensure that sixteen-bit storage elements can be used to store any intermediate values of half-pixels. In particular, when fifteen possible sub-pixel locations are defined for every pixel location, one of the specific half-pixel values may need to be generated based on other half-pixel values (i.e., requiring two levels of half-pixel interpolation), and this specific half-pixel value may require rounding to ensure that sixteen-bit storage elements can be used to store all interpolated values. |



Nokia Exhibit 2011, p. 15
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0038:<br><br>In another aspect of this disclosure, interpolation techniques are described that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. The techniques that eliminate or significantly reduce intermediate rounding may be used with or without the techniques mentioned above that use pre-defined interpolation filters. In other words, the techniques that eliminate or significantly reduce intermediate rounding may be used during interpolation according to one or more pre-defined interpolation filters, but could also be used with interpolation that uses a fixed interpolation filter or possibly with adaptive refinement interpolations techniques.<br><br>Karczewicz II, ¶0053:<br><br>Also, according to another aspect of this disclosure, video encoder 22 and video decoder 28 may apply interpolation techniques that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. In this case, video encoder 22 and video decoder 28 may round the interpolated half-pixel values purposes of half-pixel interpolation. For quarter-pixel interpolation, however, video encoder 22 and video decoder 28 may store and use non-rounded versions of the half-pixel values in order to reduce or eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding may be applied to one particular half-pixel value that requires two levels of interpolation in order to ensure that a fixed size storage elements (e.g., sixteen-bit registers) can be used to store any intermediate values. In some cases, the impact of the slight rounding on the final accuracy can be made to be zero by performing the rounding properly, as described herein. |



Nokia Exhibit 2011, p. 16
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0059:<br><br>Also, according to another aspect of this disclosure, video encoder 22 and video decoder 28 may apply interpolation techniques that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. In this case, video encoder 22 and video decoder 28 may round the interpolated half-pixel values for purposes of half-pixel interpolation. For quarter-pixel interpolation, however, video encoder 22 and video decoder 28 may store and use non-rounded versions of the half-pixel values in order to reduce or eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding may be applied to one particular half-pixel value that requires two levels of interpolation in order to ensure that fixed size storage elements (e.g., 16-bit registers) can be used to store any intermediate values.<br><br>Karczewicz II, ¶¶0093-0094:<br><br>A sub-pixel motion vector refers to a sub-pixel position in a reference picture which needs to be interpolated. H.264 defines one interpolation process for sub-pixels in which sub-pixels b and h (see FIGS. 4A-4D) may be calculated by horizontal and vertical filtering with a 6-tap filer having tap values (1, −5, 20, 20, −5, 1) as follows:<br><br>b1=C1−5*C2+20*C3+20*C4−5*C5+C6<br><br>where “C1,” “C2,” “C3,” “C4,” “C5” and “C6” represent the six closest integer pixels that surround “b” in the horizontal direction, with pixels “C3” and “C4” being the closest, “C2” and “C5” being the next closest, and “C1” and “C6” being the next closest.<br><br>h1=A3−5*B3+20*C3+20*D3−5*E3+F3 |



Nokia Exhibit 2011, p. 17
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | where "A3," "B3," "C3," "D3," "E3" and "F3" represent the six closest integer pixels that surround "h" in the vertical direction, with pixels "C3" and "D3" being the closest, "B3" and "E3" being the next closest, and "A3" and "F3" being the next closest.<br><br>In this case, the values of "b" and "h" may be defined as:<br><br>b=max(0, min(255, (b1+16)>>5))<br><br>h=max(0, min(255, (h1+16)>>5))<br>where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation.<br><br>Karczewicz II, ¶0096:<br><br>The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding.<br><br>TABLE 1<br><br>a = (C3 + b + 1) >> 1<br>c = (C4 + b + 1) >> 1<br>d = (C3 + h + 1) >> 1<br>l = (D3 + h + 1) >> 1<br>f = (j + b + 1) >> 1<br>i = (j + h + 1) >> 1<br>k = (j + ee + 1) >> 1<br>n = (j + hh + 1) >> 1 |



Nokia Exhibit 2011, p. 18
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶¶0099-0100:<br><br>In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions "b," "h," "j," "ee" and "hh," with upward rounding.<br><br>TABLE 3<br><br>a = (C3 << 5 + b1 + 32) >> 6<br>c = (C4 << 5 + b1 + 32) >> 6<br>d = (C3 << 5 + h1 + 32) >> 6<br>l = (D3 << 5 + h1 + 32) >> 6<br>f = (j1 >> 5 + b1 + 32) >> 6<br>i = (j1 >> 5 + h1 + 32) >> 6<br>k = (j1 >> 5 + ee1 + 32) >> 6<br>n = (j1 >> 5 + hh1 + 32) >> 6<br><br>In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein. |



Nokia Exhibit 2011, p. 19
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0103:<br><br>Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 20
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |

Karczewicz II, ¶¶0097-0098:

> The pixels at quarter-pixel positions labeled as “e,” “g,” “m,” and “o” are calculated according to Table 2 by averaging the two nearest pixels at half-pixel positions in the diagonal direction and then applying upward rounding.

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|

TABLE 2

```
e = (b + h + 1) >> 1
g = (b + ee + 1) >> 1
m = (h + hh + 1) >> 1
o = (ee + hh + 1) >> 1
```

The final values of those quarter-pixels may be finally clipped to values in the interval (0,255).

Karczewicz II, ¶¶0101-0102:

The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" may be calculated according to Table 4 by averaging the two nearest pixels at half-pixel positions in the diagonal direction with upward rounding.

TABLE 4

```
e = (b1 + h1 + 32) >> 6
g = (b1 + ee1 + 32) >> 6
m = (h1 + hh1 + 32) >> 6
o = (ee1 + hh1 + 32) >> 6
```

The final values of those quarter-pixels may be clipped to values in the range (0,255). By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate. In particular, half-pixel values at locations "b," "h," "ee," and "hh" may be



Nokia Exhibit 2011, p. 22
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation.<br><br>Karczewicz II, ¶0103:<br><br>Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 23
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 6

positions {e, g, m, o} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = y1 | r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y1 | −5100 | 21420 | 16s |
| r1 = r1 + 32 | r1 is y0 + y1 + 32 | −5068 | 21452 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + y1 + 32) >> 6 | −79 | 335 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 335 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Karczewicz II, ¶0095:

> To interpolate sub-pixel "j," an intermediate value "j1" is first derived as:
>
> j1=aa1−5*bb1+20*b1+20*hh1−5*ii1+jj1,
> where the intermediate values denoted as "aa1," "bb1", "hh1," "ii1" and "jj1" are derived by applying the 6-tap filter horizontally in the same manner as the calculation of b1 at the positions of "aa," "bb," "hh," "ii" and "jj." The final value j may be calculated as:



Nokia Exhibit 2011, p. 24
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | j=max(0, min(255, (j1+512)>>10)).<br><br>Karczewicz II, ¶0104-0106:<br><br>Table 7, below, illustrates a sixteen-bit implementation of deriving pixel value at pixel location "j." All 1-D half-pixels "aa1," "bb1," "b1," "hh1," "ii1" and "jj1" are obtained without any shift down. The clipping range to avoid overflow is (−2372,18640). Again shaded rows in the table indicate operations that are only performed at the end of the interpolation process, and these results are not used for the interpolation of other sub-pixel positions. The intermediate value "j1" will be saved for the interpolation of "f," "i," "k" and "n." |



Nokia Exhibit 2011, p. 25
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 7

position {j} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | y0 is 1-D half-pixel aa1 before shifting down | -2550 | 10710 | 15s |
| r2 = y5 | y5 is 1-D half-pixel jj1 before shifting down | -2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y5 | -5100 | 21420 | 16s |
| r2 = y1 | y1 is 1-D half-pixel bb1 before shifting down | -2550 | 10710 | 15s |
| r3 = y4 | y4 is 1-D half-pixel ii1 before shifting down | -2550 | 10710 | 15s |
| r2 = r2 + r3 | r2 is y1 + y4 | -5100 | 21420 | 16s |
| r1 = r1 - r2 | r1 is (y0 + y5) - (y1 + y4) | -26520 | 26520 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - (y1 + y4)) >> 2 | -6630 | 6630 | 14s |
| r3 = y2 | y2 is 1-D half-pixel b1 before shifting down | -2550 | 10710 | 15s |
| r4 = y3 | y3 is 1-D half-pixel hh1 before shifting down | -2550 | 10710 | 15s |
| r3 = r3 + r4 | r3 is y2 + y3 | -5100 | 21420 | 16s |
| r3 = max(-2372, r3) | clip r3 to avoid overflow (low side) | -2372 | 21420 | 16s |
| r3 = min(18640, r3) | clip r3 to avoid overflow (high side) | -2372 | 18640 | 16s |
| r2 = r3 - r4 | r2 is (y2 + y3) - (y1 + y4) | -23792 | 23740 | 16s |
| r1 = r1 + r2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 2 | -30422 | 30370 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 4 | -7605 | 7592 | 14s |
| r1 = r1 + r3 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3)) >> 4 | -9977 | 26232 | 16s |
| j1 = r1 | store result | | | |
| r1 = r1 + 32 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 4 | -9945 | 26264 | 16s |
| r1 = r1 >> 5 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 10 | -153 | 410 | 10s |
| r1 = max(0, r1) | clip r1 on the low side | 0 | 410 | 9u |
| r1 = min(255, r1) | clip r1 on the high side | 0 | 255 | 8u |



Nokia Exhibit 2011, p. 26
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1." |

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |



Nokia Exhibit 2011, p. 27
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 8-continued

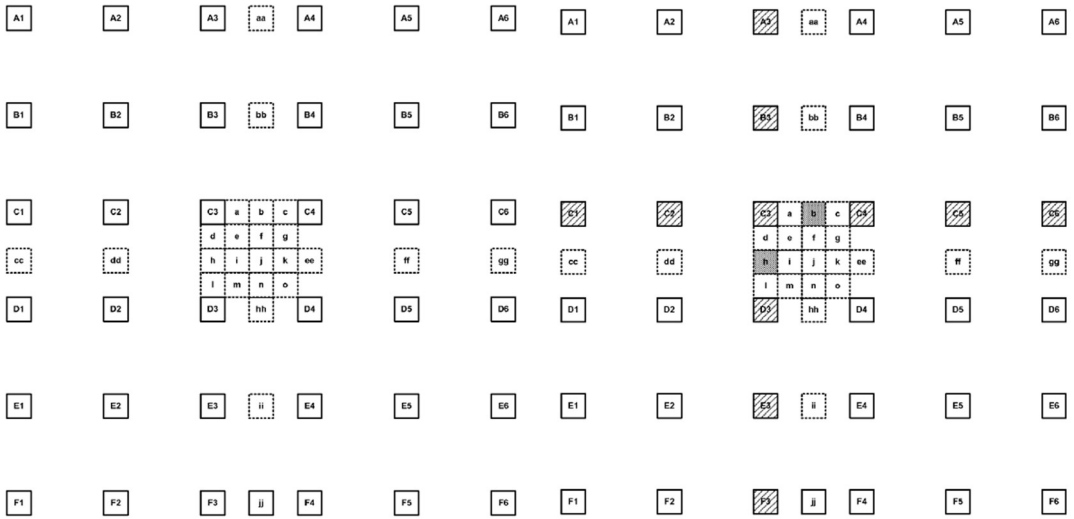
positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

In short, the discussion above demonstrates a complete technique for interpolating every possible sub-pixel location “a,” “b,” “c,” “d,” “e,” “f,” “g,” “h,” “i,” “j,” “k,” “l,” “m,” “n,” and “o” without exceeding sixteen-bit storage elements, which is desirable for an implementation. Most intermediate half-pixel values remain unrounded for purposes of quarter-pixel interpolation. However, 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. In accordance with this disclosure, however, the rounding in pixel location “j” will not impact the accuracy of the final value of quarter-pixels which rely on “j.”

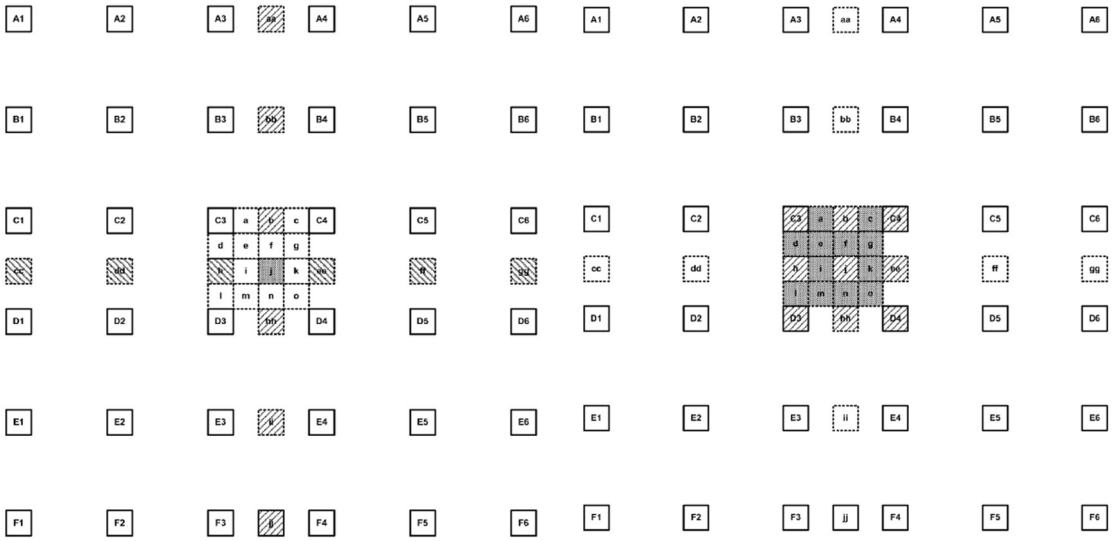


Nokia Exhibit 2011, p. 28
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, Figs. 4A-4D:<br>  <br>FIG. 4A<br><br>FIG. 4B |



Nokia Exhibit 2011, p. 29
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  <br>FIG. 4C FIG. 4D<br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 1[c] using said second reference block to obtain a second prediction, said second | Karczewicz II, as evidenced by the example citations below, discloses using the second reference block to obtain a second prediction, the second prediction having the second precision. |



Nokia Exhibit 2011, p. 30
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| prediction having the second precision; | *See, e.g.*,<br><br>Karczewicz II, ¶0010:<br><br>> In addition, this disclosure also recognizes coding inefficiencies due to conventional rounding of half-pixel values, and provides techniques that may improve interpolation by reducing or eliminating intermediate rounding. In this case, interpolated half-pixel values may be rounded for purposes of half-pixel interpolation. Quarter-pixel values, however, which may be generated based on one or more of the interpolated half-pixel values, may rely on non-rounded versions of the half-pixel values. This can eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding without sacrificing the accuracy of the final values may be applied to one specific half-pixel value in order to ensure that sixteen-bit storage elements can be used to store any intermediate values of half-pixels. In particular, when fifteen possible sub-pixel locations are defined for every pixel location, one of the specific half-pixel values may need to be generated based on other half-pixel values (i.e., requiring two levels of half-pixel interpolation), and this specific half-pixel value may require rounding to ensure that sixteen-bit storage elements can be used to store all interpolated values.<br><br>Karczewicz II, ¶0038:<br><br>> In another aspect of this disclosure, interpolation techniques are described that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. The techniques that eliminate or significantly reduce intermediate rounding may be used with or without the techniques mentioned above that use pre-defined interpolation filters. In other words, the techniques that eliminate or significantly reduce intermediate rounding may be used during interpolation according to one or more pre-defined interpolation filters, but could also be used with interpolation that uses a fixed interpolation filter or possibly with adaptive |



Nokia Exhibit 2011, p. 31
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | refinement interpolations techniques.<br><br>Karczewicz II, ¶0053:<br><br>Also, according to another aspect of this disclosure, video encoder 22 and video decoder 28 may apply interpolation techniques that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. In this case, video encoder 22 and video decoder 28 may round the interpolated half-pixel values purposes of half-pixel interpolation. For quarter-pixel interpolation, however, video encoder 22 and video decoder 28 may store and use non-rounded versions of the half-pixel values in order to reduce or eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding may be applied to one particular half-pixel value that requires two levels of interpolation in order to ensure that a fixed size storage elements (e.g., sixteen-bit registers) can be used to store any intermediate values. In some cases, the impact of the slight rounding on the final accuracy can be made to be zero by performing the rounding properly, as described herein.<br><br>Karczewicz II, ¶0059:<br><br>Also, according to another aspect of this disclosure, video encoder 22 and video decoder 28 may apply interpolation techniques that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. In this case, video encoder 22 and video decoder 28 may round the interpolated half-pixel values for purposes of half-pixel interpolation. For quarter-pixel interpolation, however, video encoder 22 and video decoder 28 may store and use non-rounded versions of the half-pixel values in order to reduce or eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding may be applied to one particular half-pixel value that requires two levels of interpolation in order to ensure that fixed size storage elements (e.g., |



Nokia Exhibit 2011, p. 32
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | 16-bit registers) can be used to store any intermediate values.<br><br>Karczewicz II, ¶¶0093-0094:<br><br>A sub-pixel motion vector refers to a sub-pixel position in a reference picture which needs to be interpolated. H.264 defines one interpolation process for sub-pixels in which sub-pixels b and h (see FIGS. 4A-4D) may be calculated by horizontal and vertical filtering with a 6-tap filer having tap values (1, −5, 20, 20, −5, 1) as follows:<br>b1=C1−5*C2+20*C3+20*C4−5*C5+C6<br>where "C1," "C2," "C3," "C4," "C5" and "C6" represent the six closest integer pixels that surround "b" in the horizontal direction, with pixels "C3" and "C4" being the closest, "C2" and "C5" being the next closest, and "C1" and "C6" being the next closest.<br>h1=A3−5*B3+20*C3+20*D3−5*E3+F3<br>where "A3," "B3," "C3," "D3," "E3" and "F3" represent the six closest integer pixels that surround "h" in the vertical direction, with pixels "C3" and "D3" being the closest, "B3" and "E3" being the next closest, and "A3" and "F3" being the next closest.<br><br>In this case, the values of "b" and "h" may be defined as:<br>b=max(0, min(255, (b1+16)>>5))<br>h=max(0, min(255, (h1+16)>>5))<br>where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation.<br><br>Karczewicz II, ¶0096:<br><br>The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding. |



Nokia Exhibit 2011, p. 33
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 1

| |
|---|
| a = (C3 + b + 1) >> 1 |
| c = (C4 + b + 1) >> 1 |
| d = (C3 + h + 1) >> 1 |
| l = (D3 + h + 1) >> 1 |
| f = (j + b + 1) >> 1 |
| i = (j + h + 1) >> 1 |
| k = (j + ee + 1) >> 1 |
| n = (j + hh + 1) >> 1 |

Karczewicz II, ¶¶0099-0100:

> In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions "b," "h," "j," "ee" and "hh," with upward rounding.



Nokia Exhibit 2011, p. 34
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|

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 |
| f = (j1 >> 5 + b1 + 32) >> 6 |
| i = (j1 >> 5 + h1 + 32) >> 6 |
| k = (j1 >> 5 + ee1 + 32) >> 6 |
| n = (j1 >> 5 + hh1 + 32) >> 6 |

> In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein.

Karczewicz II, ¶0103:

> Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed



Nokia Exhibit 2011, p. 35
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 36
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |



Nokia Exhibit 2011, p. 37
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 6

positions {e, g, m, o} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = y1 | r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y1 | −5100 | 21420 | 16s |
| r1 = r1 + 32 | r1 is y0 + y1 + 32 | −5068 | 21452 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + y1 + 32) >> 6 | −79 | 335 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 335 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Karczewicz II, ¶¶0097-0098:

> The pixels at quarter-pixel positions labeled as “e,” “g,” “m,” and “o” are calculated according to Table 2 by averaging the two nearest pixels at half-pixel positions in the diagonal direction and then applying upward rounding.

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 2

| |
|---|
| e = (b + h + 1) >> 1 |
| g = (b + ee + 1) >> 1 |
| m = (h + hh + 1) >> 1 |
| o = (ee + hh + 1) >> 1 |

The final values of those quarter-pixels may be finally clipped to values in the interval (0,255).

Karczewicz II, ¶¶0101-0102:

The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" may be calculated according to Table 4 by averaging the two nearest pixels at half-pixel positions in the diagonal direction with upward rounding.

TABLE 4

| |
|---|
| e = (b1 + h1 + 32) >> 6 |
| g = (b1 + ee1 + 32) >> 6 |
| m = (h1 + hh1 + 32) >> 6 |
| o = (ee1 + hh1 + 32) >> 6 |

The final values of those quarter-pixels may be clipped to values in the range (0,255). By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate. In particular, half-pixel values at locations "b," "h," "ee," and "hh" may be



Nokia Exhibit 2011, p. 39
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation.<br><br>Karczewicz II, ¶0095:<br><br>To interpolate sub-pixel "j," an intermediate value "j1" is first derived as:<br>j1=aa1−5*bb1+20*b1+20*hh1−5*ii1+jj1,<br>where the intermediate values denoted as "aa1," "bb1", "hh1," "ii1" and "jj1" are derived by applying the 6-tap filter horizontally in the same manner as the calculation of b1 at the positions of "aa," "bb," "hh," "ii" and "jj." The final value j may be calculated as:<br>j=max(0, min(255, (j1+512)>>10)).<br><br>Karczewicz II, ¶0104-106:<br><br>Table 7, below, illustrates a sixteen-bit implementation of deriving pixel value at pixel location "j." All 1-D half-pixels "aa1," "bb1," "b1," "hh1," "ii1" and "jj1" are obtained without any shift down. The clipping range to avoid overflow is (−2372,18640). Again shaded rows in the table indicate operations that are only performed at the end of the interpolation process, and these results are not used for the interpolation of other sub-pixel positions. The intermediate value "j1" will be saved for the interpolation of "f," "i," "k" and "n." |



Nokia Exhibit 2011, p. 40
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 7

position {j} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | y0 is 1-D half-pixel aa1 before shifting down | -2550 | 10710 | 15s |
| r2 = y5 | y5 is 1-D half-pixel jj1 before shifting down | -2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y5 | -5100 | 21420 | 16s |
| r2 = y1 | y1 is 1-D half-pixel bb1 before shifting down | -2550 | 10710 | 15s |
| r3 = y4 | y4 is 1-D half-pixel ii1 before shifting down | -2550 | 10710 | 15s |
| r2 = r2 + r3 | r2 is y1 + y4 | -5100 | 21420 | 16s |
| r1 = r1 - r2 | r1 is (y0 + y5) - (y1 + y4) | -26520 | 26520 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - (y1 + y4)) >> 2 | -6630 | 6630 | 14s |
| r3 = y2 | y2 is 1-D half-pixel b1 before shifting down | -2550 | 10710 | 15s |
| r4 = y3 | y3 is 1-D half-pixel hh1 before shifting down | -2550 | 10710 | 15s |
| r3 = r3 + r4 | r3 is y2 + y3 | -5100 | 21420 | 16s |
| r3 = max(-2372, r3) | clip r3 to avoid overflow (low side) | -2372 | 21420 | 16s |
| r3 = min(18640, r3) | clip r3 to avoid overflow (high side) | -2372 | 18640 | 16s |
| r2 = r3 - r4 | r2 is (y2 + y3) - (y1 + y4) | -23792 | 23740 | 16s |
| r1 = r1 + r2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 2 | -30422 | 30370 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 4 | -7605 | 7592 | 14s |
| r1 = r1 + r3 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3)) >> 4 | -9977 | 26232 | 16s |
| j1 = r1 | store result | | | |
| r1 = r1 + 32 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 4 | -9945 | 26264 | 16s |
| r1 = r1 >> 6 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 10 | -155 | 410 | 10s |
| r1 = max(0, r1) | clip r1 on the low side | 0 | 410 | 9u |
| r1 = min(255, r1) | clip r1 on the high side | 0 | 255 | 8u |



Nokia Exhibit 2011, p. 41
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1."

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |

TABLE 8-continued

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

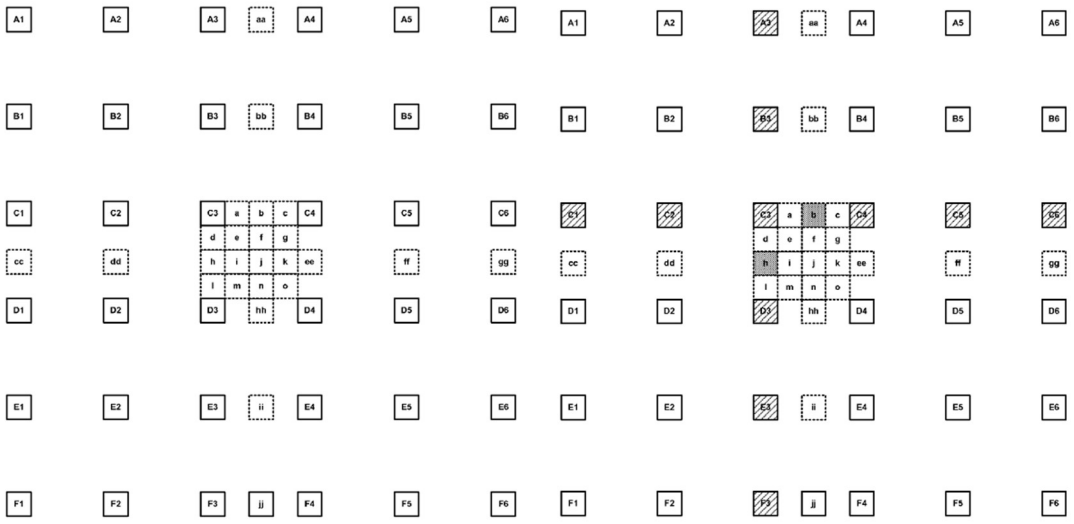


Nokia Exhibit 2011, p. 42
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | In short, the discussion above demonstrates a complete technique for interpolating every possible sub-pixel location "a," "b," "c," "d," "e," "f," "g," "h," "i," "j," "k," "l," "m," "n," and "o" without exceeding sixteen-bit storage elements, which is desirable for an implementation. Most intermediate half-pixel values remain unrounded for purposes of quarter-pixel interpolation. However, 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. In accordance with this disclosure, however, the rounding in pixel location "j" will not impact the accuracy of the final value of quarter-pixels which rely on "j."<br><br>Karczewicz II, Figs. 4A-4D: |

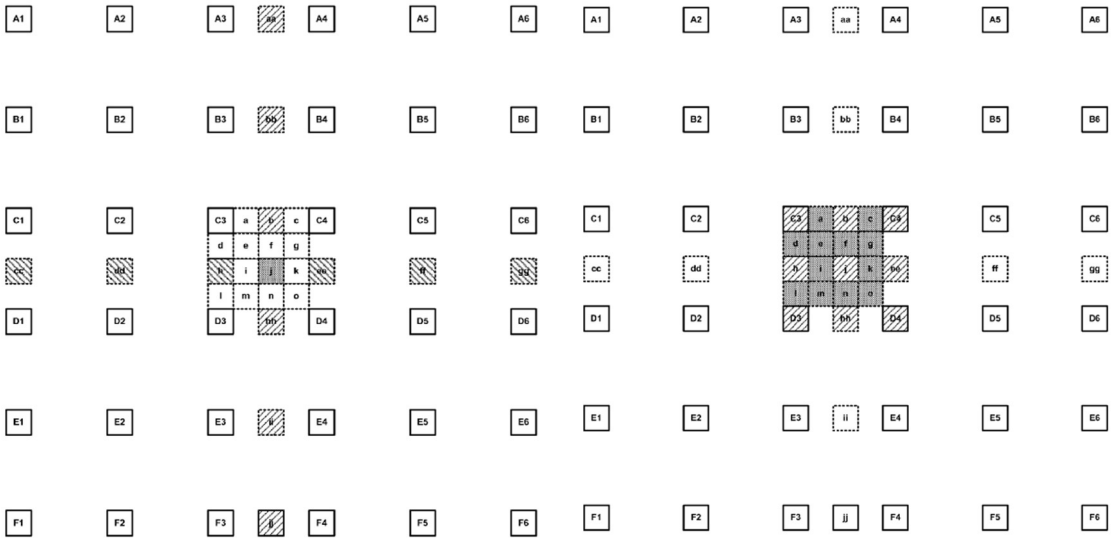


Nokia Exhibit 2011, p. 43
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| |  <br>FIG. 4A FIG. 4B |



Nokia Exhibit 2011, p. 44
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  <br>FIG. 4C<br><br>FIG. 4D<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 1[d] obtaining a combined prediction based at least partly upon said first prediction and | Karczewicz II, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction. |



Nokia Exhibit 2011, p. 45
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| said second prediction; | *See, e.g.*,<br><br>Karczewicz II, ¶¶0093-0106:<br><br>A sub-pixel motion vector refers to a sub-pixel position in a reference picture which needs to be interpolated. H.264 defines one interpolation process for sub-pixels in which sub-pixels b and h (see FIGS. 4A-4D) may be calculated by horizontal and vertical filtering with a 6-tap filer having tap values (1, −5, 20, 20, −5, 1) as follows:<br><br>b1=C1−5*C2+20*C3+20*C4−5*C5+C6<br>where “C1,” “C2,” “C3,” “C4,” “C5” and “C6” represent the six closest integer pixels that surround “b” in the horizontal direction, with pixels “C3” and “C4” being the closest, “C2” and “C5” being the next closest, and “C1” and “C6” being the next closest.<br><br>h1=A3−5*B3+20*C3+20*D3−5*E3+F3<br>where “A3,” “B3,” “C3,” “D3,” “E3” and “F3” represent the six closest integer pixels that surround “h” in the vertical direction, with pixels “C3” and “D3” being the closest, “B3” and “E3” being the next closest, and “A3” and “F3” being the next closest.<br><br>In this case, the values of “b” and “h” may be defined as:<br><br>b=max(0, min(255, (b1+16)>>5))<br><br>h=max(0, min(255, (h1+16)>>5))<br>where “>>” is a right-shift operation. In this disclosure, “>>” represents a right shift operation and “<<” represents a left shift operation. |



Nokia Exhibit 2011, p. 46
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | To interpolate sub-pixel "j," an intermediate value "j1" is first derived as:<br>j1=aa1−5*bb1+20*b1+20*hh1−5*ii1+jj1,<br>where the intermediate values denoted as "aa1," "bb1", "hh1," "ii1" and "jj1" are derived by applying the 6-tap filter horizontally in the same manner as the calculation of b1 at the positions of "aa," "bb," "hh," "ii" and "jj." The final value j may be calculated as:<br>j=max(0, min(255, (j1+512)>>10)).<br><br>The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding.<br><br>TABLE 1<br>a = (C3 + b + 1) >> 1<br>c = (C4 + b + 1) >> 1<br>d = (C3 + h + 1) >> 1<br>l = (D3 + h + 1) >> 1<br>f = (j + b + 1) >> 1<br>i = (j + h + 1) >> 1<br>k = (j + cc + 1) >> 1<br>n = (j + hh + 1) >> 1<br><br>The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" are calculated according to Table 2 by averaging the two nearest pixels at half-pixel positions in the diagonal direction and then applying upward rounding.<br><br>TABLE 2<br>e = (b + h + 1) >> 1<br>g = (b + cc + 1) >> 1<br>m = (h + hh + 1) >> 1<br>o = (cc + hh + 1) >> 1 |



Nokia Exhibit 2011, p. 47
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | The final values of those quarter-pixels may be finally clipped to values in the interval (0,255).<br><br>In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions "b," "h," "j," "ee" and "hh," with upward rounding.<br><br>TABLE 3<br>a = (C3 << 5 + b1 + 32) >> 6<br>c = (C4 << 5 + b1 + 32) >> 6<br>d = (C3 << 5 + h1 + 32) >> 6<br>l = (D3 << 5 + h1 + 32) >> 6<br>f = (j1 >> 5 + b1 + 32) >> 6<br>i = (j1 >> 5 + h1 + 32) >> 6<br>k = (j1 >> 5 + ee1 + 32) >> 6<br>n = (j1 >> 5 + hh1 + 32) >> 6<br><br>In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein.<br><br>The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" may be calculated according to Table 4 by averaging the two nearest pixels at half-pixel positions in the diagonal direction with upward rounding. |



Nokia Exhibit 2011, p. 48
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 4

| |
|---|
| e = (b1 + h1 + 32) >> 6 |
| g = (b1 + ee1 + 32) >> 6 |
| m = (h1 + hh1 + 32) >> 6 |
| o = (ee1 + hh1 + 32) >> 6 |

The final values of those quarter-pixels may be clipped to values in the range (0,255). By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate. In particular, half-pixel values at locations "b," "h," "ee," and "hh" may be unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation.

Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions.



Nokia Exhibit 2011, p. 49
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |



Nokia Exhibit 2011, p. 50
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 6

positions {e, g, m, o} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = y1 | r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y1 | −5100 | 21420 | 16s |
| r1 = r1 + 32 | r1 is y0 + y1 + 32 | −5068 | 21452 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + y1 + 32) >> 6 | −79 | 335 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 335 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Table 7, below, illustrates a sixteen-bit implementation of deriving pixel value at pixel location "j." All 1-D half-pixels "aa1," "bb1," "b1," "hh1," "ii1" and "jj1" are obtained without any shift down. The clipping range to avoid overflow is (−2372,18640). Again shaded rows in the table indicate operations that are only performed at the end of the interpolation process, and these results are not used for the interpolation of other sub-pixel positions. The intermediate value "j1" will be saved for the interpolation of "f," "i," "k" and "n."



Nokia Exhibit 2011, p. 51
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 7

position {j} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | y0 is 1-D half-pixel aa1 before shifting down | -2550 | 10710 | 15s |
| r2 = y5 | y5 is 1-D half-pixel jj1 before shifting down | -2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y5 | -5100 | 21420 | 16s |
| r2 = y1 | y1 is 1-D half-pixel bb1 before shifting down | -2550 | 10710 | 15s |
| r3 = y4 | y4 is 1-D half-pixel ii1 before shifting down | -2550 | 10710 | 15s |
| r2 = r2 + r3 | r2 is y1 + y4 | -5100 | 21420 | 16s |
| r1 = r1 - r2 | r1 is (y0 + y5) - (y1 + y4) | -26520 | 26520 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - (y1 + y4)) >> 2 | -6630 | 6630 | 14s |
| r3 = y2 | y2 is 1-D half-pixel b1 before shifting down | -2550 | 10710 | 15s |
| r4 = y3 | y3 is 1-D half-pixel hh1 before shifting down | -2550 | 10710 | 15s |
| r3 = r3 + r4 | r3 is y2 + y3 | -5100 | 21420 | 16s |
| r3 = max(-2372, r3) | clip r3 to avoid overflow (low side) | -2372 | 21420 | 16s |
| r3 = min(18640, r3) | clip r3 to avoid overflow (high side) | -2372 | 18640 | 16s |
| r2 = r3 - r4 | r2 is (y2 + y3) - (y1 + y4) | -23792 | 23740 | 16s |
| r1 = r1 + r2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 2 | -30422 | 30370 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 4 | -7605 | 7592 | 14s |
| r1 = r1 + r3 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3)) >> 4 | -9977 | 26232 | 16s |
| j1 = r1 | store result | | | |
| r1 = r1 + 32 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 4 | -9945 | 26264 | 16s |
| r1 = r1 >> 6 | r1 is ((y0 + y5) - 5* (y1 + y4)) 20* (y2 + y3) + 512) >> 10 | -155 | 410 | 10s |
| r1 = max(0, r1) | clip r1 on the low side | 0 | 410 | 9u |
| r1 = min(255, r1) | clip r1 on the high side | 0 | 255 | 8u |



Nokia Exhibit 2011, p. 52
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1."

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |

TABLE 8-continued

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

In short, the discussion above demonstrates a complete technique for interpolating every possible sub-pixel location "a," "b," "c," "d," "e," "f," "g," "h," "i," "j," "k," "l," "m," "n," and "o" without exceeding sixteen-bit storage elements, which is desirable for an implementation. Most intermediate half-pixel values remain unrounded for purposes of quarter-pixel interpolation. However, pixel location "j" is a special case that may require



Nokia Exhibit 2011, p. 53
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | rounding of intermediate results for purposes of generating its half-pixel value since pixel location "j" requires two levels of half-pixel interpolation. In accordance with this disclosure, however, the rounding in pixel location "j" will not impact the accuracy of the final value of quarter-pixels which rely on "j."<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 1[e] decreasing a precision of said combined prediction by shifting bits of the combined prediction to the right; and | Karczewicz II, as evidenced by the example citations below, discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right.<br><br>*See, e.g.*,<br><br>Karczewicz II, ¶0094:<br><br>In this case, the values of "b" and "h" may be defined as:<br><br>b=max(0, min(255, (b1+16)>>5))<br><br>h=max(0, min(255, (h1+16)>>5))<br>where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation.<br><br>*See also, e.g.*, ¶¶0094-0103 (using the ">>" right shift operation).<br><br>In this case, the values of "b" and "h" may be defined as: |



Nokia Exhibit 2011, p. 54
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | b=max(0, min(255, (b1+16)>>5))<br>h=max(0, min(255, (h1+16)>>5))<br>where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation.<br><br>To interpolate sub-pixel "j," an intermediate value "j1" is first derived as:<br>j1=aa1−5*bb1+20*b1+20*hh1−5*ii1+jj1,<br>where the intermediate values denoted as "aa1," "bb1", "hh1," "ii1" and "jj1" are derived by applying the 6-tap filter horizontally in the same manner as the calculation of b1 at the positions of "aa," "bb," "hh," "ii" and "jj." The final value j may be calculated as:<br>j=max(0, min(255, (j1+512)>>10)).<br><br>The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding.<br><br>TABLE 1<br>a = (C3 + b + 1) >> 1<br>c = (C4 + b + 1) >> 1<br>d = (C3 + h + 1) >> 1<br>l = (D3 + h + 1) >> 1<br>f = (j + b + 1) >> 1<br>i = (j + h + 1) >> 1<br>k = (j + cc + 1) >> 1<br>n = (j + hh + 1) >> 1<br><br>The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" are calculated according to Table 2 by averaging the two nearest pixels at half-pixel positions in the diagonal direction and then applying upward rounding. |



Nokia Exhibit 2011, p. 55
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 2

```
e = (b + h + 1) >> 1
g = (b + cc + 1) >> 1
m = (h + hh + 1) >> 1
o = (cc + hh + 1) >> 1
```

The final values of those quarter-pixels may be finally clipped to values in the interval (0,255).

In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as “a,” “c,” “d,” “l,” “f,” “i,” “k,” and “n” are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions “b,” “h,” “j,” “ee” and “hh,” with upward rounding.

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
f = (j1 >> 5 + b1 + 32) >> 6
i = (j1 >> 5 + h1 + 32) >> 6
k = (j1 >> 5 + ee1 + 32) >> 6
n = (j1 >> 5 + hh1 + 32) >> 6
```

In this case, the intermediate values denoted as “ee1” and “hh1” are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of “h1” and “b1” listed above, but at the positions of “ee” and “hh.” Rounding may be avoided at this stage in the generation of “ee1” and “hh1” as described herein.



Nokia Exhibit 2011, p. 56
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" may be calculated according to Table 4 by averaging the two nearest pixels at half-pixel positions in the diagonal direction with upward rounding.<br><br>TABLE 4<br><br>e = (b1 + h1 + 32) >> 6<br>g = (b1 + ee1 + 32) >> 6<br>m = (h1 + hh1 + 32) >> 6<br>o = (ee1 + hh1 + 32) >> 6<br><br>The final values of those quarter-pixels may be clipped to values in the range (0,255). By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate. In particular, half-pixel values at locations "b," "h," "ee," and "hh" may be unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation.<br><br>Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 57
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |



Nokia Exhibit 2011, p. 58
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | TABLE 6<br>positions {e, g, m, o} of FIGS. 4A-4D<br>Operation / Comment / Min value / Max value / Register size<br>r1 = y0 — r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) — −2550 — 10710 — 15s<br>r2 = y1 — r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) — −2550 — 10710 — 15s<br>r1 = r1 + r2 — r1 is y0 + y1 — −5100 — 21420 — 16s<br>r1 = r1 + 32 — r1 is y0 + y1 + 32 — −5068 — 21452 — 16s<br>r1 = r1 >> 6 — r1 is (y0 + y1 + 32) >> 6 — −79 — 335 — 11s<br>r1 = max (0, r1) — clip r1 on the low side — 0 — 335 — 10u<br>r1 = min (255, r1) — clip r1 on the high side — 0 — 255 — 8u<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 1[f] 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. | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.*,<br><br>Karczewicz II, ¶0004:<br><br>Block based inter-coding is a very useful coding technique that relies on temporal prediction |



Nokia Exhibit 2011, p. 59
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | to reduce or remove temporal redundancy between video blocks of successive coded units of a video sequence. The coded units may comprise video frames, slices of video frames, groups of pictures, or another defined unit of video blocks. For inter-coding, the video encoder performs motion estimation and motion compensation to track the movement of corresponding video blocks of two or more adjacent coded units. Motion estimation generates motion vectors, which indicate the displacement of video blocks relative to corresponding prediction video blocks in one or more reference frames or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the reference frame or other coded unit. After motion compensation, residual video blocks are formed by subtracting prediction video blocks from the original video blocks being coded.<br><br>Karczewicz II, ¶0006:<br><br>A coded video block may be represented by prediction information that can be used to create or identify a predictive block, and a residual block of data indicative of differences between the block being coded and the predictive block. The prediction information may comprise the one or more motion vectors that are used to identify the predictive block of data. Given the motion vectors, the decoder is able to reconstruct the predictive blocks that were used to code the residual. Thus, given a set of residual blocks and a set of motion vectors (and possibly some additional syntax), the decoder may be able to reconstruct a video frame that was originally encoded. Inter-coding based on motion estimation and motion compensation can achieve very good compression because successive video frames or other types of coded units are often very similar. An encoded video sequence may comprise blocks of residual data, motion vectors, and possibly other types of syntax.<br><br>Karczewicz II, ¶0035:<br><br>This disclosure describes various interpolation techniques performed by an encoder and a |



Nokia Exhibit 2011, p. 60
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | decoder during the motion compensation process of video coding. According to one aspect of this disclosure, the encoder may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data. The interpolated prediction data that achieves the highest levels of compression may be selected at the encoder, and the interpolation filter that was used may be coded as syntax and communicated to the decoding device as part of the coded video data. The interpolated prediction data comprises reference data. Video data to be coded (e.g., a video block) may be subtracted from interpolated prediction data (e.g., an interpolated predictive reference block) in order to define a residual block of encoded data, which may then be transformed, quantized and entropy coded.<br><br>Karczewicz II, ¶0050:<br><br>Smaller video blocks can provide better resolution, and may be used for locations of a video frame that include high levels of detail. In general, macroblocks (MBs) and the various sub-blocks may be considered to be video blocks. In addition, a slice may be considered to be a series of video blocks, such as MBs and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Video encoder 22 and video decoder 28 perform inter-based predictive coding, which involves the generation of predictive reference data and the subtraction of a video block to be coded from the predictive reference data to generate residual data, which may then be transformed, quantized and entropy coded. The inter-based predictive coding may include interpolation of the predictive data in accordance with this disclosure.<br><br>Karczewicz II, ¶0058:<br><br>Once motion estimation unit 32 has selected the motion vector for the video block to be coded, motion compensation unit 35 generates the predictive video block associated with that motion vector. According to this disclosure, however, motion compensation unit 35 may |



Nokia Exhibit 2011, p. 61
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | consider several versions of any predictive video block that has sub-pixel resolution. In this case, motion compensation unit 35 may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data for the video block to be coded. Motion compensation unit 35 then selects the interpolated prediction data (e.g., an interpolated video block associated with one of the interpolation filters) that achieves the highest levels of compression. The interpolation filter that was used to generate the interpolated data may be coded as interpolation syntax and communicated to entropy coding unit 46 for inclusion in the coded bitstream. Once motion compensation unit 35 has selected and applied the best interpolation filter, the motion compensation unit 35 generates the predictive data using that interpolation filter, and video encoder applies adder 48 to subtract that predictive data from the video block being coded to generate residual data.<br><br>Karczewicz II, ¶0060:<br><br>As noted, once motion compensation unit 35 generated prediction data (e.g., an interpolated predictive video block), video encoder 50 forms a residual video block by subtracting the prediction data from the original video block being coded. Adder 48 represents the component or components that perform this subtraction operation. Transform unit 38 applies a transform, such as a discrete cosine transform (DCT) or a conceptually similar transform, to the residual block, producing a video block comprising residual transform block coefficients. Transform unit 38, for example, may perform other transforms, such as those defined by the H.264 standard, which are conceptually similar to DCT. Wavelet transforms, integer transforms, sub-band transforms or other types of transforms could also be used. In any case, transform unit 38 applies the transform to the residual block, producing a block of residual transform coefficients. The transform may convert the residual information from a pixel domain to a frequency domain. |



Nokia Exhibit 2011, p. 62
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0073:<br><br>After motion estimation, the best motion vector for a given video block may be identified, e.g., possibly using a rate-distortion model to balance the coding rate and quality. Then, the prediction video block is formed during motion compensation using the best motion vector. As outlined above, the residual video block is formed by subtracting the prediction video block from the original video block. A transform is then applied on the residual block, and the transform coefficients are quantized and entropy coded to further reduce bit rate.<br><br>Karczewicz II, ¶0005:<br><br>The video encoder may also apply transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of residual blocks. Transform techniques may comprise discrete cosine transforms (DCTs) or conceptually similar processes. Alternatively, wavelet transforms, integer transforms, or other types of transforms may be used. In a DCT process, as an example, a set of pixel values are converted into transform coefficients, which may represent the energy of the pixel values in the frequency domain. Quantization is applied to the transform coefficients, and generally involves a process that limits the number of bits associated with any given transform coefficient. Entropy coding comprises one or more processes that collectively compress a sequence of quantized transform coefficients. Examples of entropy coding include but are not limited to content adaptive variable length coding (CAVLC) and context adaptive binary arithmetic coding (CABAC).<br><br>Karczewicz II, ¶0061: |



Nokia Exhibit 2011, p. 63
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | As noted, once motion compensation unit 35 generated prediction data (e.g., an interpolated predictive video block), video encoder 50 forms a residual video block by subtracting the prediction data from the original video block being coded. Adder 48 represents the component or components that perform this subtraction operation. Transform unit 38 applies a transform, such as a discrete cosine transform (DCT) or a conceptually similar transform, to the residual block, producing a video block comprising residual transform block coefficients. Transform unit 38, for example, may perform other transforms, such as those defined by the H.264 standard, which are conceptually similar to DCT. Wavelet transforms, integer transforms, sub-band transforms or other types of transforms could also be used. In any case, transform unit 38 applies the transform to the residual block, producing a block of residual transform coefficients. The transform may convert the residual information from a pixel domain to a frequency domain.<br><br>Karczewicz II, ¶0110:<br><br>FIG. 6 is a flow diagram consistent with one or more aspects of video decoding consistent with this disclosure. FIG. 6 will be discussed from the perspective of video encoder 50 shown in FIG. 2. In particular, video encoder 60 receives an encoded video data (111), and receives a syntax element that identifies an interpolation filter from a plurality of interpolation filters (112). The encoded bitstream may include both the encoded residual video data and the syntax element that identifies the interpolation filter that was used at the encoder. Entropy decoding unit 52 may entropy decode the received bitstream to parse out the transformed and quantized residual blocks, which are sent to inverse transform quantization unit 56 and inverse transform unit 58, and to parse out the interpolation syntax element and motion vectors, which are sent to motion compensation unit 55.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination* |



Nokia Exhibit 2011, p. 64
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the first motion vector points to a subpixel, the first prediction is obtained by interpolation using pixel values of the first reference block.<br><br>*See, e.g.*,<br><br>Karczewicz II, ¶0007:<br><br>> Interpolation techniques have been developed in order to improve the level of compression that can be achieved in inter-coding. In this case, the predictive data generated during motion compensation, which is used to code a video block, may be interpolated from the pixels of video blocks of the video frame or other coded unit used in motion estimation. Interpolation is often performed to generate predictive half-pixel values (half-pel) and predictive quarter-pixel values (quarter-pel). Such interpolation often generates predictive blocks that are even more similar to the video blocks being coded than the actual video blocks of the predictive frame or other coded unit used in the video coding.<br><br>Karczewicz II, ¶0010:<br><br>> In addition, this disclosure also recognizes coding inefficiencies due to conventional rounding of half-pixel values, and provides techniques that may improve interpolation by reducing or eliminating intermediate rounding. In this case, interpolated half-pixel values may be rounded for purposes of half-pixel interpolation. Quarter-pixel values, however, which may be generated based on one or more of the interpolated half-pixel values, may rely on non-rounded versions of the half-pixel values. This can eliminate propagation of rounding |



Nokia Exhibit 2011, p. 65
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding without sacrificing the accuracy of the final values may be applied to one specific half-pixel value in order to ensure that sixteen-bit storage elements can be used to store any intermediate values of half-pixels. In particular, when fifteen possible sub-pixel locations are defined for every pixel location, one of the specific half-pixel values may need to be generated based on other half-pixel values (i.e., requiring two levels of half-pixel interpolation), and this specific half-pixel value may require rounding to ensure that sixteen-bit storage elements can be used to store all interpolated values.<br><br>Karczewicz II, ¶0042:<br><br>The illustrated system 10 of FIG. 1 is merely exemplary. The interpolation techniques of this disclosure may be performed by any encoding device that supports motion compensated interpolation to sub-pixel resolution. Source device 12 and destination device 16 are merely examples of such coding devices. In this case, source device 12 generates coded video data for transmission to destination device 16. Devices 12, 16 may operate in a substantially symmetrical manner such that each of devices 12, 16 include video encoding and decoding components, e.g., in a combined encoder-decoder (CODEC). Hence, system 10 may support one-way or two-way video transmission between video devices 12, 16, e.g., for video streaming, video playback, video broadcasting, or video telephony.<br><br>Karczewicz II, ¶0053:<br><br>Also, according to another aspect of this disclosure, video encoder 22 and video decoder 28 may apply interpolation techniques that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. In this case, video encoder 22 and video decoder 28 may round the interpolated half-pixel values purposes of half-pixel interpolation. For quarter-pixel interpolation, however, video encoder 22 and video decoder |



Nokia Exhibit 2011, p. 66
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | 28 may store and use non-rounded versions of the half-pixel values in order to reduce or eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding may be applied to one particular half-pixel value that requires two levels of interpolation in order to ensure that a fixed size storage elements (e.g., sixteen-bit registers) can be used to store any intermediate values. In some cases, the impact of the slight rounding on the final accuracy can be made to be zero by performing the rounding properly, as described herein.<br><br>Karczewicz II, ¶0059:<br><br>Also, according to another aspect of this disclosure, video encoder 22 and video decoder 28 may apply interpolation techniques that eliminate or significantly reduce intermediate rounding of half-pixel values for quarter-pixel interpolation. In this case, video encoder 22 and video decoder 28 may round the interpolated half-pixel values for purposes of half-pixel interpolation. For quarter-pixel interpolation, however, video encoder 22 and video decoder 28 may store and use non-rounded versions of the half-pixel values in order to reduce or eliminate propagation of rounding inaccuracies from the half-pixel values to the quarter-pixel values. In some cases, slight rounding may be applied to one particular half-pixel value that requires two levels of interpolation in order to ensure that fixed size storage elements (e.g., 16-bit registers) can be used to store any intermediate values.<br><br>Karczewicz II, ¶0066:<br><br>Again, the techniques of this disclosure concern motion compensated interpolation in which pixel values of predictive video blocks are interpolated to sub-pixel resolution. The encoder uses the techniques of this disclosure to identify a desirable interpolation filter from a plurality of pre-defined interpolation filters. The different filters may be characterized by different sets of filter coefficients, different numbers of filter coefficients, or different filter |



Nokia Exhibit 2011, p. 67
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | types. The decoder interprets syntax elements sent from the encoder in order to identify the same desirable set of interpolation filter coefficients used by the encoder.<br><br>Karczewicz II, ¶0067:<br><br>FIGS. 4A-4D are conceptual diagrams illustrating integer pixels and interpolated pixels that may comprise pixels of predictive video blocks. In the conceptual illustration of FIG. 4, the different boxes represent pixels. Capitalized letters (in the boxes with solid lines) represent integer pixel locations, while small letters (in the boxes with dotted lines) represent the interpolated pixel locations. The lettered labels may be used herein to describe the pixel locations, or pixel positions, or may refer to the pixel values associated with the various locations. Pixel locations "aa," "bb," "cc," "dd," "ee," "ff," "gg," "hh," "ii," and "jj," are half-pixel locations used in the fractional interpolations of various fractional locations associated with pixel location "C3."<br><br>Karczewicz II, ¶0069:<br><br>In the ITU H.264/AVC standard, in order to obtain luma signals at half-pixel positions, a 6-tap Wiener filter with coefficients [1, −5, 20, 20, −5, 1] is typically used. Then, in order to obtain luma signals at quarter-pixel locations, a bilinear filter is used. The bilinear filter may also be used in fractional pixel interpolation for the chroma components, which may have up to ⅛-pixel precision in H.264/AVC.<br><br>Karczewicz II, ¶0070:<br><br>Half-pixel interpolations of pixel locations "b" and "h" are demonstrated in FIG. 4B in the horizontal and vertical directions respectively. In this case, pixel location "b" may be interpolated based on integer pixels "C1," "C2," C3," "C4," "C5," and "C6." Similarly, pixel |



Nokia Exhibit 2011, p. 68
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | location “h” may be interpolated based on integer pixels “A3,” “B3,” C3,” “D3,” “E3,” and “F3.”Different interpolation filters (e.g., different sets of filter taps) may be applied to generate different interpolated values of pixel locations “b” and “h” as described herein. In FIG. 4B, interpolated pixel locations “b” and “h” are shown with shading, and integer pixel locations “C1,”“C2,” C3,” “C4,” “C5,” “C6,” “A3”, “B3”, “C3”, “D3”, “E3” and “F3” are shown with cross-hashes.<br><br>Karczewicz II, ¶¶0093-0106:<br><br>A sub-pixel motion vector refers to a sub-pixel position in a reference picture which needs to be interpolated. H.264 defines one interpolation process for sub-pixels in which sub-pixels b and h (see FIGS. 4A-4D) may be calculated by horizontal and vertical filtering with a 6-tap filer having tap values (1, −5, 20, 20, −5, 1) as follows:<br><br>b1=C1−5*C2+20*C3+20*C4−5*C5+C6<br>where “C1,” “C2,” “C3,” “C4,” “C5” and “C6” represent the six closest integer pixels that surround “b” in the horizontal direction, with pixels “C3” and “C4” being the closest, “C2” and “C5” being the next closest, and “C1” and “C6” being the next closest.<br><br>h1=A3−5*B3+20*C3+20*D3−5*E3+F3<br>where “A3,” “B3,” “C3,” “D3,” “E3” and “F3” represent the six closest integer pixels that surround “h” in the vertical direction, with pixels “C3” and “D3” being the closest, “B3” and “E3” being the next closest, and “A3” and “F3” being the next closest.<br><br>In this case, the values of “b” and “h” may be defined as:<br>b=max(0, min(255, (b1+16)>>5)) |



Nokia Exhibit 2011, p. 69
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | h=max(0, min(255, (h1+16)>>5))<br>where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation.<br><br>To interpolate sub-pixel "j," an intermediate value "j1" is first derived as:<br>j1=aa1−5*bb1+20*b1+20*hh1−5*ii1+jj1,<br>where the intermediate values denoted as "aa1," "bb1", "hh1," "ii1" and "jj1" are derived by applying the 6-tap filter horizontally in the same manner as the calculation of b1 at the positions of "aa," "bb," "hh," "ii" and "jj." The final value j may be calculated as:<br>j=max(0, min(255, (j1+512)>>10)).<br><br>The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding.<br><br>TABLE 1<br>a = (C3 + b + 1) >> 1<br>c = (C4 + b + 1) >> 1<br>d = (C3 + h + 1) >> 1<br>l = (D3 + h + 1) >> 1<br>f = (j + b + 1) >> 1<br>i = (j + h + 1) >> 1<br>k = (j + cc + 1) >> 1<br>n = (j + hh + 1) >> 1<br><br>The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" are calculated according to Table 2 by averaging the two nearest pixels at half-pixel positions in the diagonal direction and then applying upward rounding. |



Nokia Exhibit 2011, p. 70
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 2

```
e = (b + h + 1) >> 1
g = (b + cc + 1) >> 1
m = (h + hh + 1) >> 1
o = (cc + hh + 1) >> 1
```

The final values of those quarter-pixels may be finally clipped to values in the interval (0,255).

In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions "b," "h," "j," "ee" and "hh," with upward rounding.

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
f = (j1 >> 5 + b1 + 32) >> 6
i = (j1 >> 5 + h1 + 32) >> 6
k = (j1 >> 5 + ee1 + 32) >> 6
n = (j1 >> 5 + hh1 + 32) >> 6
```

In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein.



Nokia Exhibit 2011, p. 71
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" may be calculated according to Table 4 by averaging the two nearest pixels at half-pixel positions in the diagonal direction with upward rounding.<br><br>TABLE 4<br>e = (b1 + h1 + 32) >> 6<br>g = (b1 + ee1 + 32) >> 6<br>m = (h1 + hh1 + 32) >> 6<br>o = (ee1 + hh1 + 32) >> 6<br><br>The final values of those quarter-pixels may be clipped to values in the range (0,255). By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate. In particular, half-pixel values at locations "b," "h," "ee," and "hh" may be unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation.<br><br>Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed |



Nokia Exhibit 2011, p. 72
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

at the end of the process, and are not used for interpolation of other sub-pixel positions.

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 |



Nokia Exhibit 2011, p. 73
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 6

positions {e, g, m, o} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = y1 | r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y1 | −5100 | 21420 | 16s |
| r1 = r1 + 32 | r1 is y0 + y1 + 32 | −5068 | 21452 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + y1 + 32) >> 6 | −79 | 335 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 335 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Table 7, below, illustrates a sixteen-bit implementation of deriving pixel value at pixel location "j." All 1-D half-pixels "aa1," "bb1," "b1," "hh1," "ii1" and "jj1" are obtained without any shift down. The clipping range to avoid overflow is (−2372,18640). Again shaded rows in the table indicate operations that are only performed at the end of the interpolation process, and these results are not used for the interpolation of other sub-pixel positions. The intermediate value "j1" will be saved for the interpolation of "f," "i," "k" and "n."



Nokia Exhibit 2011, p. 74
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 7

position {j} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | y0 is 1-D half-pixel aa1 before shifting down | -2550 | 10710 | 15s |
| r2 = y5 | y5 is 1-D half-pixel jj1 before shifting down | -2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y5 | -5100 | 21420 | 16s |
| r2 = y1 | y1 is 1-D half-pixel bb1 before shifting down | -2550 | 10710 | 15s |
| r3 = y4 | y4 is 1-D half-pixel ii1 before shifting down | -2550 | 10710 | 15s |
| r2 = r2 + r3 | r2 is y1 + y4 | -5100 | 21420 | 16s |
| r1 = r1 - r2 | r1 is (y0 + y5) - (y1 + y4) | -26520 | 26520 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - (y1 + y4)) >> 2 | -6630 | 6630 | 14s |
| r3 = y2 | y2 is 1-D half-pixel b1 before shifting down | -2550 | 10710 | 15s |
| r4 = y3 | y3 is 1-D half-pixel hh1 before shifting down | -2550 | 10710 | 15s |
| r3 = r3 + r4 | r3 is y2 + y3 | -5100 | 21420 | 16s |
| r3 = max(-2372, r3) | clip r3 to avoid overflow (low side) | -2372 | 21420 | 16s |
| r3 = min(18640, r3) | clip r3 to avoid overflow (high side) | -2372 | 18640 | 16s |
| r2 = r3 - r4 | r2 is (y2 + y3) - (y1 + y4) | -23792 | 23740 | 16s |
| r1 = r1 + r2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 2 | -30422 | 30370 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 4 | -7605 | 7592 | 14s |
| r1 = r1 + r3 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3)) >> 4 | -9977 | 26232 | 16s |
| j1 = r1 | store result | | | |
| r1 = r1 + 32 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 4 | -9945 | 26264 | 16s |
| r1 = r1 >> 6 | r1 is ((y0 + y5) - 5* (y1 + y4)) 20* (y2 + y3) + 512) >> 10 | -155 | 410 | 10s |
| r1 = max(0, r1) | clip r1 on the low side | 0 | 410 | 9u |
| r1 = min(255, r1) | clip r1 on the high side | 0 | 255 | 8u |



Nokia Exhibit 2011, p. 75
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1."

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |

TABLE 8-continued

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

In short, the discussion above demonstrates a complete technique for interpolating every possible sub-pixel location "a," "b," "c," "d," "e," "f," "g," "h," "i," "j," "k," "l," "m," "n," and "o" without exceeding sixteen-bit storage elements, which is desirable for an implementation. Most intermediate half-pixel values remain unrounded for purposes of quarter-pixel interpolation. However, pixel location "j" is a special case that may require



Nokia Exhibit 2011, p. 76
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | rounding of intermediate results for purposes of generating its half-pixel value since pixel location "j" requires two levels of half-pixel interpolation. In accordance with this disclosure, however, the rounding in pixel location "j" will not impact the accuracy of the final value of quarter-pixels which rely on "j."<br><br>Karczewicz II, ¶0067:<br><br>FIGS. 4A-4D are conceptual diagrams illustrating integer pixels and interpolated pixels that may comprise pixels of predictive video blocks. In the conceptual illustration of FIG. 4, the different boxes represent pixels. Capitalized letters (in the boxes with solid lines) represent integer pixel locations, while small letters (in the boxes with dotted lines) represent the interpolated pixel locations. The lettered labels may be used herein to describe the pixel locations, or pixel positions, or may refer to the pixel values associated with the various locations. Pixel locations "aa," "bb," "cc," "dd," "ee," "ff," "gg," "hh," "ii," and "jj," are half-pixel locations used in the fractional interpolations of various fractional locations associated with pixel location "C3." |



Nokia Exhibit 2011, p. 77
Snap/Hisense v. Nokia
IPR2025-01375

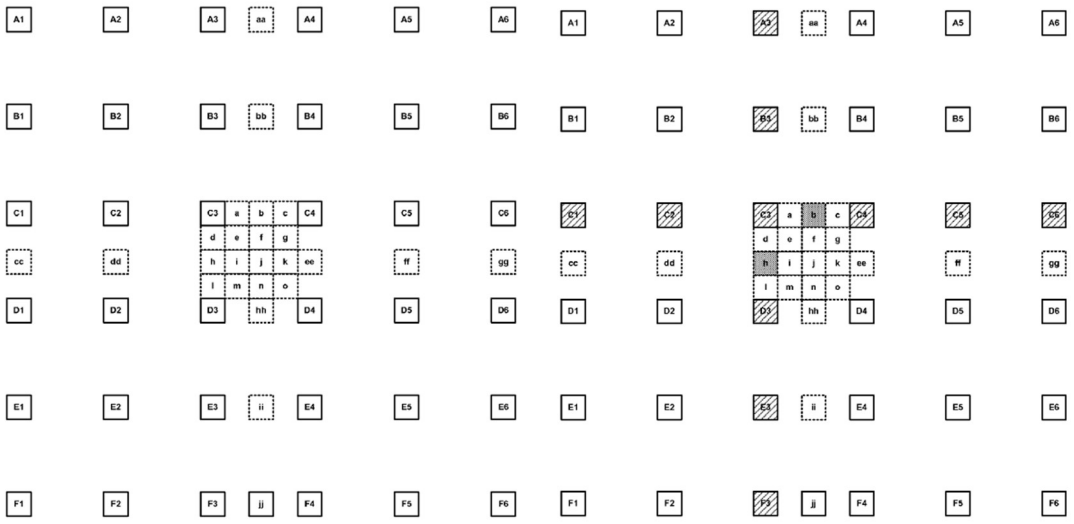
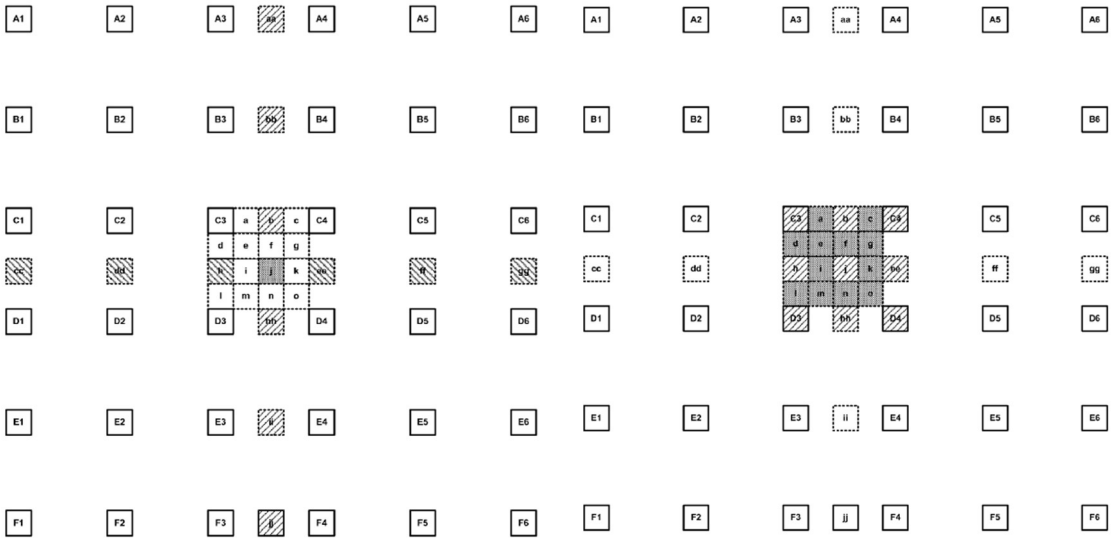
| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  |

FIG. 4A

FIG. 4B



Nokia Exhibit 2011, p. 78
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  <br>FIG. 4C FIG. 4D<br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 3. The method according to claim 2, wherein said first prediction is obtained by | Karczewicz II, as evidenced by the example citations below, discloses that the first prediction is obtained by interpolation using values of the first reference block by right shifting a sum of a P-tap |



Nokia Exhibit 2011, p. 79
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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. | filter using values of the first reference block.<br><br>*See, e.g.,*<br><br>Karczewicz II, ¶0095:<br><br>To interpolate sub-pixel "j," an intermediate value "j1" is first derived as: j1=aa1−5*bb1+20*b1+20*hh1−5*ii1+jj1, where the intermediate values denoted as "aa1," "bb1", "hh1," "ii1" and "jj1" are derived by applying the 6-tap filter horizontally in the same manner as the calculation of b1 at the positions of "aa," "bb," "hh," "ii" and "jj." The final value j may be calculated as: j=max(0, min(255, (j1+512)>>10)).<br><br>Karczewicz II, ¶0067:<br><br>FIGS. 4A-4D are conceptual diagrams illustrating integer pixels and interpolated pixels that may comprise pixels of predictive video blocks. In the conceptual illustration of FIG. 4, the different boxes represent pixels. Capitalized letters (in the boxes with solid lines) represent integer pixel locations, while small letters (in the boxes with dotted lines) represent the interpolated pixel locations. The lettered labels may be used herein to describe the pixel locations, or pixel positions, or may refer to the pixel values associated with the various locations. Pixel locations "aa," "bb," "cc," "dd," "ee," "ff," "gg," "hh," "ii," and "jj," are half-pixel locations used in the fractional interpolations of various fractional locations associated with pixel location "C3." |



Nokia Exhibit 2011, p. 80
Snap/Hisense v. Nokia
IPR2025-01375

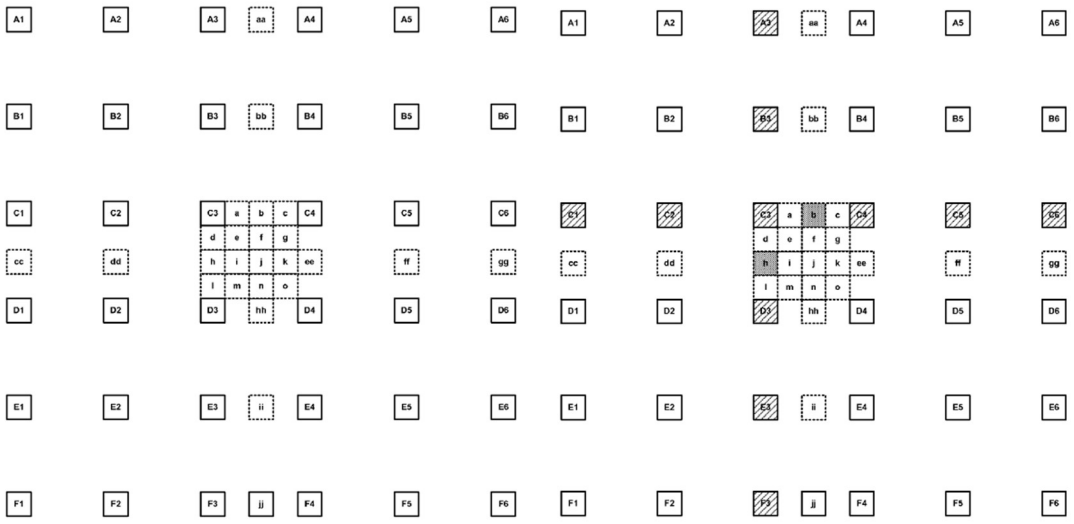
| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  |

FIG. 4A

FIG. 4B



Nokia Exhibit 2011, p. 81
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|

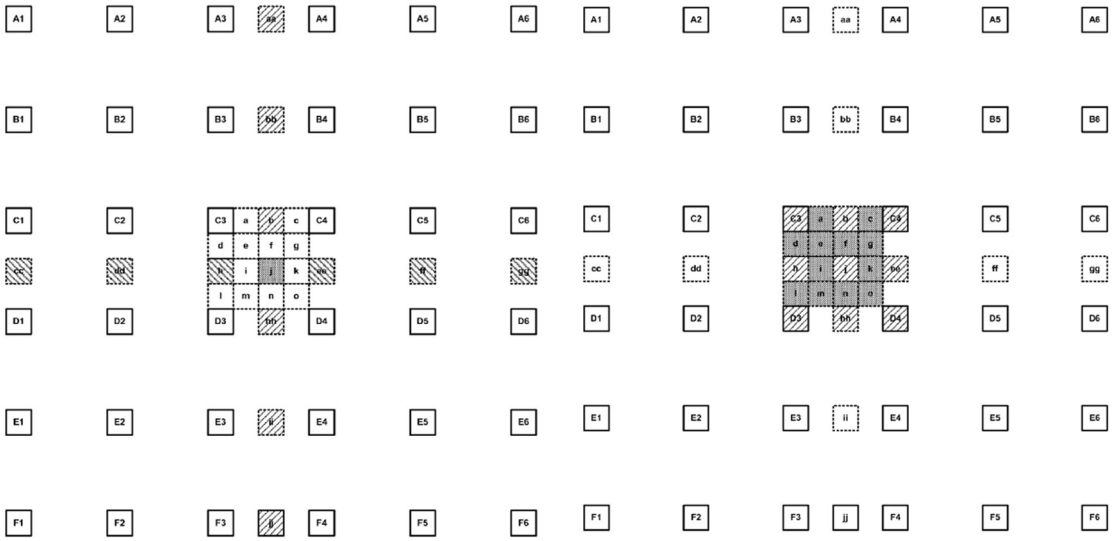


FIG. 4C



FIG. 4D

Karczewicz II, ¶¶0099-0100:

> In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as “a,” “c,” “d,” “l,” “f,” “i,” “k,” and “n” are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions “b,” “h,” “j,” “ee” and “hh,” with upward rounding.



Nokia Exhibit 2011, p. 82
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|

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 |
| f = (j1 >> 5 + b1 + 32) >> 6 |
| i = (j1 >> 5 + h1 + 32) >> 6 |
| k = (j1 >> 5 + ee1 + 32) >> 6 |
| n = (j1 >> 5 + hh1 + 32) >> 6 |

> In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein.

Karczewicz II, ¶0105:

> Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1."



Nokia Exhibit 2011, p. 83
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |

TABLE 8-continued

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |



Nokia Exhibit 2011, p. 84
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the second motion vector points to an integer sample, the second prediction is obtained by shifting values of the second reference block to the left.<br><br>*See, e.g.,*<br><br>Karczewicz II, ¶0096:<br><br>> The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding.<br><br>TABLE 1<br><br>a = (C3 + b + 1) >> 1<br>c = (C4 + b + 1) >> 1<br>d = (C3 + h + 1) >> 1<br>l = (D3 + h + 1) >> 1<br>f = (j + b + 1) >> 1<br>i = (j + h + 1) >> 1<br>k = (j + ee + 1) >> 1<br>n = (j + hh + 1) >> 1 |



Nokia Exhibit 2011, p. 85
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶¶0099-0100:<br><br>In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions "b," "h," "j," "ee" and "hh," with upward rounding.<br><br>TABLE 3<br><br>a = (C3 << 5 + b1 + 32) >> 6<br>c = (C4 << 5 + b1 + 32) >> 6<br>d = (C3 << 5 + h1 + 32) >> 6<br>l = (D3 << 5 + h1 + 32) >> 6<br>f = (j1 >> 5 + b1 + 32) >> 6<br>i = (j1 >> 5 + h1 + 32) >> 6<br>k = (j1 >> 5 + ee1 + 32) >> 6<br>n = (j1 >> 5 + hh1 + 32) >> 6<br><br>In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein.<br><br>Karczewicz II, ¶0093:<br><br>A sub-pixel motion vector refers to a sub-pixel position in a reference picture which needs to be interpolated. H.264 defines one interpolation process for sub-pixels in which sub-pixels b |



Nokia Exhibit 2011, p. 86
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | and h (see FIGS. 4A-4D) may be calculated by horizontal and vertical filtering with a 6-tap filer having tap values (1, −5, 20, 20, −5, 1) as follows:<br>b1=C1−5*C2+20*C3+20*C4−5*C5+C6<br>where "C1," "C2," "C3," "C4," "C5" and "C6" represent the six closest integer pixels that surround "b" in the horizontal direction, with pixels "C3" and "C4" being the closest, "C2" and "C5" being the next closest, and "C1" and "C6" being the next closest.<br>h1=A3−5*B3+20*C3+20*D3−5*E3+F3<br>where "A3," "B3," "C3," "D3," "E3" and "F3" represent the six closest integer pixels that surround "h" in the vertical direction, with pixels "C3" and "D3" being the closest, "B3" and "E3" being the next closest, and "A3" and "F3" being the next closest.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the decreasing the precision of the combined prediction by shifting bits of the combined prediction to the right, further comprises inserting a rounding offset to the combined prediction before the decreasing.<br><br>*See, e.g.,*<br><br>Karczewicz II, ¶¶0096-0106:<br><br>The pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" may be derived according to Table 1 by averaging the two nearest pixels at integer and half-pixel positions and then applying upward rounding. |



Nokia Exhibit 2011, p. 87
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 1

| |
|---|
| a = (C3 + b + 1) >> 1 |
| c = (C4 + b + 1) >> 1 |
| d = (C3 + h + 1) >> 1 |
| l = (D3 + h + 1) >> 1 |
| f = (j + b + 1) >> 1 |
| i = (j + h + 1) >> 1 |
| k = (j + ee + 1) >> 1 |
| n = (j + hh + 1) >> 1 |

The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" are calculated according to Table 2 by averaging the two nearest pixels at half-pixel positions in the diagonal direction and then applying upward rounding.

TABLE 2

| |
|---|
| e = (b + h + 1) >> 1 |
| g = (b + ee + 1) >> 1 |
| m = (h + hh + 1) >> 1 |
| o = (ee + hh + 1) >> 1 |

The final values of those quarter-pixels may be finally clipped to values in the interval (0,255).

In order to keep the highest possible precision through the intermediate steps, any shifting, rounding and clipping operations may be avoided until the very last step of the interpolation process. Unlike in the H.264 standard defined implementation, the pixels at quarter-pixel positions labeled as "a," "c," "d," "l," "f," "i," "k," and "n" are derived according to Table 3 by averaging the two nearest pixels at integer and half-pixel positions "b," "h," "j," "ee" and "hh," with upward rounding.



Nokia Exhibit 2011, p. 88
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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
f = (j1 >> 5 + b1 + 32) >> 6
i = (j1 >> 5 + h1 + 32) >> 6
k = (j1 >> 5 + ee1 + 32) >> 6
n = (j1 >> 5 + hh1 + 32) >> 6
```

In this case, the intermediate values denoted as "ee1" and "hh1" are derived by applying the 6-tap filter vertically and horizontally in the same manner as the calculation of "h1" and "b1" listed above, but at the positions of "ee" and "hh." Rounding may be avoided at this stage in the generation of "ee1" and "hh1" as described herein.

The pixels at quarter-pixel positions labeled as "e," "g," "m," and "o" may be calculated according to Table 4 by averaging the two nearest pixels at half-pixel positions in the diagonal direction with upward rounding.

TABLE 4

```
e = (b1 + h1 + 32) >> 6
g = (b1 + ee1 + 32) >> 6
m = (h1 + hh1 + 32) >> 6
o = (ee1 + hh1 + 32) >> 6
```

The final values of those quarter-pixels may be clipped to values in the range (0,255). By preserving the full precision of the intermediate values, the interpolated sub-pixels will be more accurate. In particular, half-pixel values at locations "b," "h," "ee," and "hh" may be unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation.



Nokia Exhibit 2011, p. 89
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 90
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 5

positions {s, 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 |



Nokia Exhibit 2011, p. 91
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 6

positions {e, g, m, o} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = y1 | r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y1 | −5100 | 21420 | 16s |
| r1 = r1 + 32 | r1 is y0 + y1 + 32 | −5068 | 21452 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + y1 + 32) >> 6 | −79 | 335 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 335 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Table 7, below, illustrates a sixteen-bit implementation of deriving pixel value at pixel location "j." All 1-D half-pixels "aa1," "bb1," "b1," "hh1," "ii1" and "jj1" are obtained without any shift down. The clipping range to avoid overflow is (−2372,18640). Again shaded rows in the table indicate operations that are only performed at the end of the interpolation process, and these results are not used for the interpolation of other sub-pixel positions. The intermediate value "j1" will be saved for the interpolation of "f," "i," "k" and "n."



Nokia Exhibit 2011, p. 92
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 7

position {j} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | y0 is 1-D half-pixel aa1 before shifting down | -2550 | 10710 | 15s |
| r2 = y5 | y5 is 1-D half-pixel jj1 before shifting down | -2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y5 | -5100 | 21420 | 16s |
| r2 = y1 | y1 is 1-D half-pixel bb1 before shifting down | -2550 | 10710 | 15s |
| r3 = y4 | y4 is 1-D half-pixel ii1 before shifting down | -2550 | 10710 | 15s |
| r2 = r2 + r3 | r2 is y1 + y4 | -5100 | 21420 | 16s |
| r1 = r1 - r2 | r1 is (y0 + y5) - (y1 + y4) | -26520 | 26520 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - (y1 + y4)) >> 2 | -6630 | 6630 | 14s |
| r3 = y2 | y2 is 1-D half-pixel b1 before shifting down | -2550 | 10710 | 15s |
| r4 = y3 | y3 is 1-D half-pixel hh1 before shifting down | -2550 | 10710 | 15s |
| r3 = r3 + r4 | r3 is y2 + y3 | -5100 | 21420 | 16s |
| r3 = max(-2372, r3) | clip r3 to avoid overflow (low side) | -2372 | 21420 | 16s |
| r3 = min(18640, r3) | clip r3 to aveid overflow (high side) | -2372 | 18640 | 16s |
| r2 = r3 - r4 | r2 is (y2 + y3) - (y1 + y4) | -23792 | 23740 | 16s |
| r1 = r1 + r2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 2 | -30422 | 30370 | 16s |
| r1 = r1 >> 2 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 4* (y2 + y3)) >> 4 | -7605 | 7592 | 14s |
| r1 = r1 + r3 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3)) >> 4 | -9977 | 26232 | 16s |
| j1 = r1 | store result | | | |
| r1 = r1 + 32 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 4 | -9945 | 26264 | 16s |
| r1 = r1 >> 6 | r1 is ((y0 + y5) - 5* (y1 + y4)) + 20* (y2 + y3) + 512) >> 10 | -155 | 410 | 10s |
| r1 = max(0, r1) | clip r1 on the low side | 0 | 410 | 9u |
| r1 = min(255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1."



Nokia Exhibit 2011, p. 93
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |

TABLE 8-continued

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

In short, the discussion above demonstrates a complete technique for interpolating every possible sub-pixel location “a,” “b,” “c,” “d,” “e,” “f,” “g,” “h,” “i,” “j,” “k,” “l,” “m,” “n,” and “o” without exceeding sixteen-bit storage elements, which is desirable for an implementation. Most intermediate half-pixel values remain unrounded for purposes of quarter-pixel interpolation. However, 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. In accordance with this disclosure, however, the rounding in pixel location “j” will not impact the accuracy of the final value of quarter-pixels which rely on “j.”



Nokia Exhibit 2011, p. 94
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that 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 the first prediction and values of the second prediction.<br><br>*See, e.g.,*<br><br>Karczewicz II, ¶0089:<br><br>> The average filter for a particular sub-pixel position may also be normalized such that all filter coefficients add up to 1.0. The average filter may also be quantized to a certain fixed-point precision (e.g., 13-bit precision). The use of fixed-point precision in the filter coefficient ensures that implementations across different platforms will not have drifts. Furthermore, a true fixed-point implementation may be derived from such pre-defined filter coefficients. Any customized filters may have different characteristics. Specifically, different filters may be non-separable filters or separable filters, and different filters may define different filter supports (e.g., 6×6 or 4×4).<br><br>Karczewicz II, ¶0103:<br><br>> Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the |



Nokia Exhibit 2011, p. 95
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 96
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |



Nokia Exhibit 2011, p. 97
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 6

positions {e, g, m, o} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = y1 | r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r1 = r1 + r2 | r1 is y0 + y1 | −5100 | 21420 | 16s |
| r1 = r1 + 32 | r1 is y0 + y1 + 32 | −5068 | 21452 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + y1 + 32) >> 6 | −79 | 335 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 335 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |

Karczewicz II, ¶0105:

> Table 8, below demonstrates steps that can be taken for sixteen-bit implementation of interpolating {f,i,k,n}, which are the positions that use to interpolate the intermediate value "j1."

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

TABLE 8

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r1 = y0 | r1 is y0 (1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) | −2550 | 10710 | 15s |
| r2 = j1 | r2 is j1 (2-D half-pixel j1 before shifting down) | −9914 | 26232 | 16s |

TABLE 8-continued

positions {f, i, k, n} of FIGS. 4A-4D

| Operation | Comment | Min value | Max value | Register size |
|---|---|---|---|---|
| r2 = r2 >> 1 | r2 is j1 >> 1 | −4957 | 13116 | 15s |
| r1 = r1 + r2 | r1 is y0 + (j1 >> 1) | −7507 | 23826 | 16s |
| r1 = r1 + 32 | r1 is y0 + (j1 >> 1) + 32 | −7491 | 23842 | 16s |
| r1 = r1 >> 6 | r1 is (y0 + (j1 >> 1) + 32) >> 6 | −235 | 745 | 11s |
| r1 = max (0, r1) | clip r1 on the low side | 0 | 745 | 10u |
| r1 = min (255, r1) | clip r1 on the high side | 0 | 255 | 8u |



Nokia Exhibit 2011, p. 99
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 7[preamble] An apparatus for encoding a block of pixels, the apparatus comprising: | Karczewicz II, as evidenced by the example citations below, discloses an apparatus for encoding a block of pixels.<br><br>*See, e.g.,*<br><br>*See limitation 1[preamble].*<br><br>Karczewicz II, ¶0003:<br><br>> Digital video capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, digital cameras, digital recording devices, video gaming devices, video game consoles, cellular or satellite radio telephones, and the like. Digital video devices implement video compression techniques, such as MPEG-2, MPEG-4, or ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), to transmit and receive digital video information more efficiently. Video compression techniques may perform spatial prediction and/or temporal prediction to reduce or remove redundancy inherent in video sequences.<br><br>Karczewicz II, ¶0015:<br><br>> In another example, this disclosure provides an apparatus that encodes video data, the apparatus comprising a video encoder that includes a motion compensation unit that generates prediction data. The motion compensation unit interpolates pixel values of reference video |



Nokia Exhibit 2011, p. 100
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | data based on a plurality of different pre-defined interpolation filters, and the video encoder encodes the video data based on the prediction data.<br><br>Karczewicz II, ¶0017:<br><br>In another example, this disclosure provides an apparatus that interpolates predictive video data for video coding, wherein the apparatus includes a motion compensation unit that generates half-pixel values based on integer pixel values, rounds the half-pixel values to generate half-pixel interpolated values, stores the half-pixel values as non-rounded versions of the half-pixel values, and generates quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0021:<br><br>In another example, this disclosure provides a device that encodes video data, the device comprising a video encoder including a motion compensation unit that generates prediction data by interpolating pixel values of reference video data based on a plurality of different pre-defined interpolation filters, and encodes the video data based on the prediction data, and a wireless transmitter that transmits the video data to another device.<br><br>Karczewicz II, ¶0023:<br><br>In another example, this disclosure provides a device that encodes video data, the device comprising a video encoder including a motion compensation unit that generates half-pixel values based on integer pixel values, rounds the half-pixel values to generate half-pixel interpolated values, stores the half-pixel values as non-rounded versions of the half-pixel values, and generates quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values, and a wireless transmitter that transmits the video data |



Nokia Exhibit 2011, p. 101
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | encoded by the motion compensation unit to another device.<br><br>Karczewicz II, ¶0025:<br><br>The techniques described in this disclosure may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the software may be executed in one or more processors, such as a microprocessor, application specific integrated circuit (ASIC), field programmable gate array (FPGA), or digital signal processor (DSP). The software that executes the techniques may be initially stored in a computer-readable medium and loaded and executed in the processor.<br><br>Karczewicz II, ¶0026:<br><br>Accordingly, this disclosure also contemplates a computer-readable storage medium comprising instructions that upon execution cause a device to encode video data, wherein the instructions cause the device to generate prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on a plurality of different pre-defined interpolation filters, and encode the video data based on the prediction data.<br><br>Karczewicz II, ¶0048:<br><br>Video encoder 22 and video decoder 28 each may be implemented as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic, software, hardware, firmware or any combinations thereof. Each of video encoder 22 and video decoder 28 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined CODEC in a respective mobile device, subscriber device, broadcast device, |

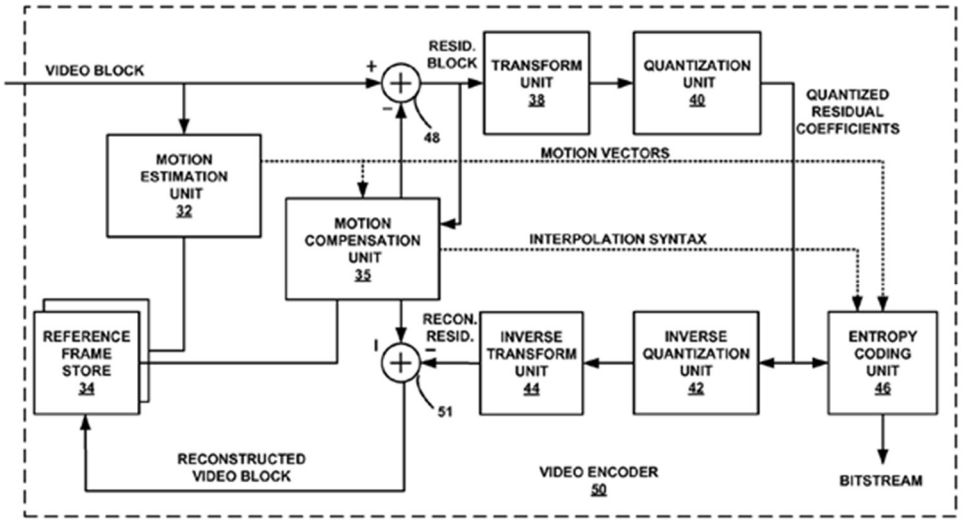


Nokia Exhibit 2011, p. 102
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | server, or the like.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 7[a] 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: | Karczewicz II, as evidenced by the example citations below, discloses at least one processor and at least one memory including computer program code.<br><br>*See, e.g.,*<br><br>Karczewicz II, ¶0004:<br><br>Block based inter-coding is a very useful coding technique that relies on temporal prediction to reduce or remove temporal redundancy between video blocks of successive coded units of a video sequence. The coded units may comprise video frames, slices of video frames, groups of pictures, or another defined unit of video blocks. For inter-coding, the video encoder performs motion estimation and motion compensation to track the movement of corresponding video blocks of two or more adjacent coded units. Motion estimation generates motion vectors, which indicate the displacement of video blocks relative to corresponding prediction video blocks in one or more reference frames or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the reference frame or other coded unit. After motion compensation, residual video blocks are formed by subtracting prediction video blocks from the original video blocks being coded. |



Nokia Exhibit 2011, p. 103
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, Fig. 2:<br>  <br><br>FIG. 2<br><br>Karczewicz II, ¶0056:<br><br>During the encoding process, video encoder 50 receives a video block to be coded, and motion estimation unit 32 and motion compensation unit 35 perform inter-predictive coding. Motion estimation unit 32 and motion compensation unit 35 may be highly integrated, but are illustrated separately for conceptual purposes. Motion estimation is typically considered the process of generating motion vectors, which estimate motion for video blocks. A motion vector, for example, may indicate the displacement of a predictive block within a predictive |



Nokia Exhibit 2011, p. 104
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | frame (or other coded unit) relative to the current block being coded within the current frame (or other coded unit). Motion compensation is typically considered the process of fetching or generating the predictive block based on the motion vector determined by motion estimation. Again, motion estimation unit 32 and motion compensation unit 35 may be functionally integrated. The interpolation techniques described in this disclosure are described as being performed by motion compensation unit 35. However, interpolation may be performed during motion estimation in order to facilitate the selection of the best motion vectors.<br><br>Karczewicz II, ¶0057:<br><br>In accordance with this disclosure, motion estimation 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of a predictive coded unit (e.g., a previous frame). At this point, motion compensation unit 35 may perform interpolation in order to generate predictive data at sub-pixel resolution. In some cases, during motion estimation, the interpolation may be based on a fixed interpolation filter. In other cases, the different interpolation filters applied during motion compensation (as outlined below) may also be used during the motion estimation process for purposes of motion vector selection.<br><br>Karczewicz II, ¶0058:<br><br>Once motion estimation unit 32 has selected the motion vector for the video block to be coded, motion compensation unit 35 generates the predictive video block associated with that motion vector. According to this disclosure, however, motion compensation unit 35 may consider several versions of any predictive video block that has sub-pixel resolution. In this case, motion compensation unit 35 may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data for the video block to be coded. Motion compensation unit 35 then selects the interpolated prediction data (e.g., an |



Nokia Exhibit 2011, p. 105
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | interpolated video block associated with one of the interpolation filters) that achieves the highest levels of compression. The interpolation filter that was used to generate the interpolated data may be coded as interpolation syntax and communicated to entropy coding unit 46 for inclusion in the coded bitstream. Once motion compensation unit 35 has selected and applied the best interpolation filter, the motion compensation unit 35 generates the predictive data using that interpolation filter, and video encoder applies adder 48 to subtract that predictive data from the video block being coded to generate residual data.<br><br>Karczewicz II, ¶0073:<br><br>After motion estimation, the best motion vector for a given video block may be identified, e.g., possibly using a rate-distortion model to balance the coding rate and quality. Then, the prediction video block is formed during motion compensation using the best motion vector. As outlined above, the residual video block is formed by subtracting the prediction video block from the original video block. A transform is then applied on the residual block, and the transform coefficients are quantized and entropy coded to further reduce bit rate.<br><br>Karczewicz II, ¶0089:<br><br>The average filter for a particular sub-pixel position may also be normalized such that all filter coefficients add up to 1.0. The average filter may also be quantized to a certain fixed-point precision (e.g., 13-bit precision). The use of fixed-point precision in the filter coefficient ensures that implementations across different platforms will not have drifts. Furthermore, a true fixed-point implementation may be derived from such pre-defined filter coefficients. Any customized filters may have different characteristics. Specifically, different filters may be non-separable filters or separable filters, and different filters may define different filter supports (e.g., 6×6 or 4×4). |



Nokia Exhibit 2011, p. 106
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0103:<br><br>Unfortunately, interpolation of value "j" can be more complicated since the interpolation at position j requires two levels of half-pixel interpolation. In particular, to interpolate "j," twenty bits may be required to represent the intermediate value "j1." In this case, the intermediate version "j1" of "j" may be rounded to fit within sixteen bits. In this case, the operations discussed above may be implemented with minor modification so that all operations can fit within sixteen bit data structures without sacrificing precision. The interpolation of "b" and "h" may be the same as that defined above, except that their intermediate versions "b1" and "h1" may be stored for other sub-pixel interpolation. The following Tables show the interpolation process for other sub-pixels in sixteen bit storage elements. In the Tables below, the operations defined in each column are performed sequentially through the respective table. Shaded rows contain operations that are performed at the end of the process, and are not used for interpolation of other sub-pixel positions. |



Nokia Exhibit 2011, p. 107
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

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 |



Nokia Exhibit 2011, p. 108
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | TABLE 6<br>positions {e, g, m, o} of FIGS. 4A-4D<br><br>Operation \| Comment \| Min value \| Max value \| Register size<br>r1 = y0 \| r1 is y0 (y0 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) \| −2550 \| 10710 \| 15s<br>r2 = y1 \| r2 is y1 (y1 is a 1-D half-pixel such as b1, h1, ee1 and hh1 before shifting down) \| −2550 \| 10710 \| 15s<br>r1 = r1 + r2 \| r1 is y0 + y1 \| −5100 \| 21420 \| 16s<br>r1 = r1 + 32 \| r1 is y0 + y1 + 32 \| −5068 \| 21452 \| 16s<br>r1 = r1 >> 6 \| r1 is (y0 + y1 + 32) >> 6 \| −79 \| 335 \| 11s<br>r1 = max (0, r1) \| clip r1 on the low side \| 0 \| 335 \| 10u<br>r1 = min (255, r1) \| clip r1 on the high side \| 0 \| 255 \| 8u<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 7[b] determine, for a current block, a first reference block based on a first motion vector and a second reference block | Karczewicz II, as evidenced by the example citations below, discloses 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 |



Nokia Exhibit 2011, p. 109
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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; | second reference block have values with a first precision.<br><br>*See, e.g.*,<br><br>*See* limitation 1[a]. |
| 7[c] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision; | Karczewicz II, as evidenced by the example citations below, discloses using the first reference block to obtain a first prediction, the first prediction having a second precision, which is higher than the first precision.<br><br>*See, e.g.,*<br><br>*See limitation 1[b].* |
| 7[d] use said second reference block to obtain a second prediction, said second prediction having the second precision; | Karczewicz II, as evidenced by the example citations below, discloses using the second reference block to obtain a second prediction, the second prediction having the second precision.<br><br>*See, e.g.,*<br><br>*See limitation 1[c].* |
| 7[e] obtain a combined prediction based at least partly upon said first prediction and said second prediction; | Karczewicz II, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction.<br><br>*See, e.g.,* |



**Nokia Exhibit 2011, p. 110**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *See limitation 1[d].* |
| 7[f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and | Karczewicz II, as evidenced by the example citations below, discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right.<br><br>*See, e.g.,*<br><br>*See limitation 1[e].* |
| 7[g] 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. | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,*<br><br>*See limitation 1[f].* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the first motion vector points to a subpixel, the first prediction is obtained by interpolation using pixel values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 2.* |
| 9. The apparatus according to | Karczewicz II, as evidenced by the example citations below, discloses that the first prediction is obtained by interpolation using values of the first reference block by right shifting a sum of a P-tap |



Nokia Exhibit 2011, p. 111
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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. | filter using values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 3.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the second motion vector points to an integer sample, the second prediction is obtained by shifting values of the second reference block to the left.<br><br>*See, e.g.,*<br><br>*See claim 4.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the at least one memory and computer code are configured to cause the apparatus to decrease the precision of the combined prediction by shifting bits of the combined prediction to the right, by inserting a rounding offset to the combined prediction before the decreasing.<br><br>*See, e.g.,*<br><br>*See claim 5.*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the* |



Nokia Exhibit 2011, p. 112
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that 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 the first prediction and values of the second prediction.<br><br>*See, e.g.,*<br><br>*See claim 6.* |
| 13[preamble] 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: | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,*<br><br>*See limitations 1[preamble], 7[preamble], and 7[a].*<br><br>Karczewicz II, ¶0028:<br><br>> In addition, this disclosure also contemplates a computer-readable storage medium comprising instructions that upon execution cause a device to interpolate predictive video data for video coding, wherein the instructions cause the device to generate half-pixel values based on integer pixel values, round the half-pixel values to generate half-pixel interpolated |



Nokia Exhibit 2011, p. 113
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | values, store the half-pixel values as non-rounded versions of the half-pixel values, and generate quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Accordingly, this disclosure also contemplates a computer-readable storage medium comprising instructions that upon execution cause a device to encode video data, wherein the instructions cause the device to generate prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on a plurality of different pre-defined interpolation filters, and encode the video data based on the prediction data. |
| 13[a] 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; | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,*<br><br>*See limitation 1[a].* |
| 13[b] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision; | Karczewicz II, as evidenced by the example citations below, discloses using the first reference block to obtain a first prediction, the first prediction having a second precision, which is higher than the first precision.<br><br>*See, e.g.,*<br><br>*See limitation 1[b].* |



Nokia Exhibit 2011, p. 114
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 13[c] use said second reference block to obtain a second prediction, said second prediction having the second precision; | Karczewicz II, as evidenced by the example citations below, discloses using the second reference block to obtain a second prediction, the second prediction having the second precision.<br><br>*See, e.g.,*<br><br>*See limitation 1[c].* |
| 13[d] obtain a combined prediction based at least partly upon said first prediction and said second prediction; | Karczewicz II, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction.<br><br>*See, e.g.,*<br><br>*See limitation 1[d].* |
| 13[e] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and | Karczewicz II, as evidenced by the example citations below, discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right.<br><br>*See, e.g.,*<br><br>*See limitation 1[e].* |
| 13[f] encode residual data in a bitstream, wherein the residual data is determined based upon a difference between the combined prediction and the block of | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,* |



**Nokia Exhibit 2011, p. 115**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| pixels. | *See limitation 1[f].* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the first motion vector points to a subpixel, the first prediction is obtained by interpolation using pixel values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 2* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the first prediction is obtained by interpolation using values of the first reference block by right shifting a sum of a P-tap filter using values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 3.* |
| 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 | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the second motion vector points to an integer sample, the second prediction is obtained by shifting values of the second reference block to the left.<br><br>*See, e.g.,*<br><br>*See claim 4.* |



**Nokia Exhibit 2011, p. 116**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the program code instructions configured to decrease the precision of the 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 the decreasing.<br><br>*See, e.g.,*<br><br>*See claims 5 and 11.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that 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 the first prediction and values of the second prediction.<br><br>*See, e.g.,*<br><br>*See claim 6.* |
| 19[preamble] A method for | Karczewicz II, as evidenced by the example citations below, discloses a method for decoding a block |



Nokia Exhibit 2011, p. 117
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| decoding a block of pixels, the method comprising: | of pixels.<br><br>*See, e.g.,*<br><br>Karczewicz II, Abstract:<br><br>> This disclosure describes various interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding. In one example, an encoder interpolates pixel values of reference video data based on a plurality of different pre-defined interpolation filters. In this example, the decoder receives a syntax element that identifies an interpolation filter, and interpolates pixel values of reference video data based on the interpolation filter identified by the syntax element. In another example, a method of interpolating predictive video data includes generating half-pixel values based on integer pixel values, rounding the half-pixel values to generate half-pixel interpolated values, storing the half-pixel values as non-rounded versions of the half-pixel values, and generating quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0002:<br><br>> This disclosure relates to digital video coding and, more particularly, fractional interpolations of predictive data used in video coding.<br><br>Karczewicz II, ¶0013:<br><br>> In another example, this disclosure provides a method of decoding video data. The decoding method comprises receiving a syntax element that identifies an interpolation filter from a plurality of different pre-defined interpolation filters, generating prediction data, wherein |



Nokia Exhibit 2011, p. 118
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | generating the prediction data includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element, and decoding the video data based on the prediction data.<br><br>Karczewicz II, ¶0014:<br><br>In another example, this disclosure provides a method of interpolating predictive video data for video coding. The method comprises generating half-pixel values based on integer pixel values, rounding the half-pixel values to generate half-pixel interpolated values, storing the half-pixel values as non-rounded versions of the half-pixel values, and generating quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0035:<br><br>This disclosure describes various interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding. According to one aspect of this disclosure, the encoder may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data. The interpolated prediction data that achieves the highest levels of compression may be selected at the encoder, and the interpolation filter that was used may be coded as syntax and communicated to the decoding device as part of the coded video data. The interpolated prediction data comprises reference data. Video data to be coded (e.g., a video block) may be subtracted from interpolated prediction data (e.g., an interpolated predictive reference block) in order to define a residual block of encoded data, which may then be transformed, quantized and entropy coded. |

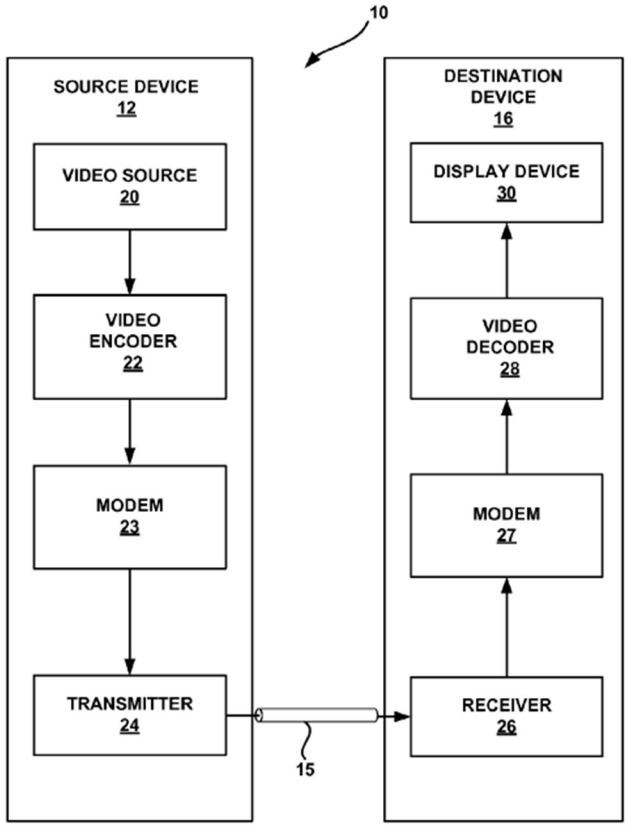


Nokia Exhibit 2011, p. 119
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, Fig.1: |



Nokia Exhibit 2011, p. 120
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| |  <br><br>FIG. 1<br>Karczewicz II, ¶0041: |



Nokia Exhibit 2011, p. 121
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | In the example of FIG. 1, source device 12 may include a video source 20, video encoder 22, a modulator/demodulator (modem) 23 and a transmitter 24. Destination device 16 may include a receiver 26 a modem 27, a video decoder 28, and a display device 30. In accordance with this disclosure, video encoder 22 of source device 12 may be configured to apply one or more of the interpolation techniques of this disclosure as part of a video encoding process. Similarly, video decoder 28 of destination device 16 may be configured to apply one or more of the interpolation techniques of this disclosure as part of a video decoding process.<br><br>Karczewicz II, Fig. 3: |



Nokia Exhibit 2011, p. 122
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

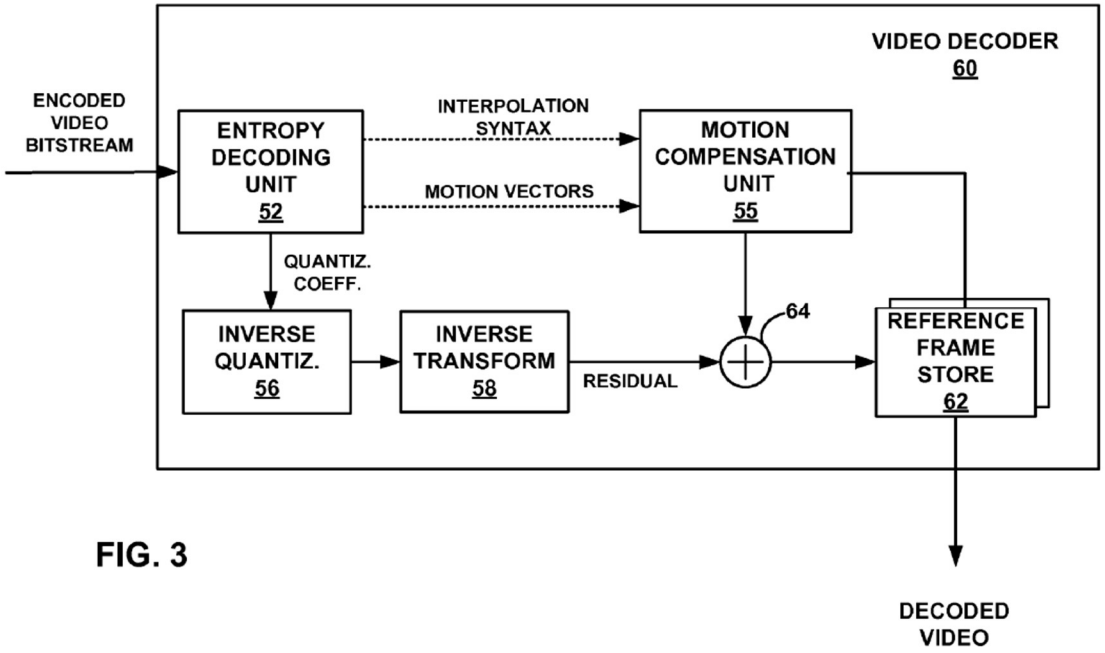



FIG. 3

Karczewicz II, ¶0049:

> A video sequence typically includes a series of video frames. Video encoder 22 operates on video blocks within individual video frames in order to encode the video data. The video blocks may have fixed or varying sizes, and may differ in size according to a specified coding standard. Each video frame includes a series of slices. Each slice may include a series of macroblocks, which may be arranged into sub-blocks. As an example, the ITU-T H.264



Nokia Exhibit 2011, p. 123
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | standard supports intra prediction in various block sizes, such as 16 by 16, 8 by 8, or 4 by 4 for luma components, and 8×8 for chroma components, as well as inter prediction in various block sizes, such as 16 by 16, 16 by 8, 8 by 16, 8 by 8, 8 by 4, 4 by 8 and 4 by 4 for luma components and corresponding scaled sizes for chroma components. Video blocks may comprise blocks of pixel data, or blocks of transformation coefficients, e.g., following a transformation process such as discrete cosine transform or a conceptually similar transformation process.<br><br>Karczewicz II, ¶0050:<br><br>Smaller video blocks can provide better resolution, and may be used for locations of a video frame that include high levels of detail. In general, macroblocks (MBs) and the various sub-blocks may be considered to be video blocks. In addition, a slice may be considered to be a series of video blocks, such as MBs and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Video encoder 22 and video decoder 28 perform inter-based predictive coding, which involves the generation of predictive reference data and the subtraction of a video block to be coded from the predictive reference data to generate residual data, which may then be transformed, quantized and entropy coded. The inter-based predictive coding may include interpolation of the predictive data in accordance with this disclosure.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 19[a] determining, for a current block, a first reference | Karczewicz II, as evidenced by the example citations below, discloses determining, for a current block, a first reference block based on a first motion vector and a second reference block based on a |



Nokia Exhibit 2011, p. 124
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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; | 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.<br><br>*See, e.g.,*<br><br>*See limitation 1[a].*<br><br>Karczewicz II, ¶0006:<br><br>> A coded video block may be represented by prediction information that can be used to create or identify a predictive block, and a residual block of data indicative of differences between the block being coded and the predictive block. The prediction information may comprise the one or more motion vectors that are used to identify the predictive block of data. Given the motion vectors, the decoder is able to reconstruct the predictive blocks that were used to code the residual. Thus, given a set of residual blocks and a set of motion vectors (and possibly some additional syntax), the decoder may be able to reconstruct a video frame that was originally encoded. Inter-coding based on motion estimation and motion compensation can achieve very good compression because successive video frames or other types of coded units are often very similar. An encoded video sequence may comprise blocks of residual data, motion vectors, and possibly other types of syntax.<br><br>Karczewicz II, ¶0063:<br><br>> FIG. 3 is a block diagram illustrating an example of a video decoder 60, which decodes a video sequence that is encoded in the manner described herein. Video decoder 60 includes a motion compensation unit 55 that performs the interpolation techniques of this disclosure for decoding. In particular, on the decoding side, motion compensation unit 55 may receive a syntax element from entropy decoding unit 52 that identifies an interpolation filter from a |



Nokia Exhibit 2011, p. 125
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | plurality of different pre-defined interpolation filters. Motion compensation unit 55 may generate prediction data, which includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element. Specifically, motion compensation unit 55 may generate the prediction data based on motion vectors received from entropy decoding unit 52 and the interpolations as defined by syntax element (labeled interpolation syntax in FIG. 3). Based on this interpolated prediction data, the video data (e.g., a reconstructed residual video block) can be decoded.<br><br>Karczewicz II, Fig. 3: |



Nokia Exhibit 2011, p. 126
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|

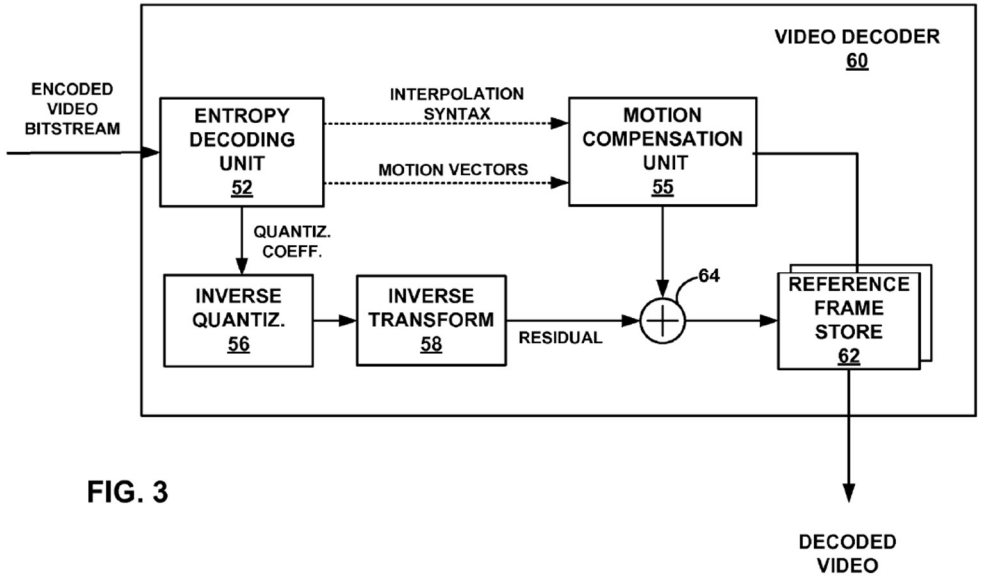



FIG. 3

Karczewicz II, ¶0064:

> Entropy decoding unit 52 entropy decodes the received bitstream to generate quantized coefficients and the syntax (e.g., interpolation syntax and motion vectors sent to motion compensation unit 55). Inverse quantization unit 56 inverse quantizes, i.e., de-quantizes, the quantized block coefficients. The inverse quantization process may be a conventional process as defined by H.264 decoding. Inverse transform unit 58 applies an inverse transform, e.g., an inverse DCT or conceptually similar inverse transform process, to the transform coefficients in order to produce residual blocks in the pixel domain. Motion compensation unit 55



Nokia Exhibit 2011, p. 127
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | produces motion compensated blocks in the manner described herein, e.g., including interpolation based on a set of interpolation filter coefficients identified by the syntax element (i.e., the interpolation syntax).<br><br>Karczewicz II, ¶0110:<br><br>FIG. 6 is a flow diagram consistent with one or more aspects of video decoding consistent with this disclosure. FIG. 6 will be discussed from the perspective of video encoder 50 shown in FIG. 2. In particular, video encoder 60 receives an encoded video data (111), and receives a syntax element that identifies an interpolation filter from a plurality of interpolation filters (112). The encoded bitstream may include both the encoded residual video data and the syntax element that identifies the interpolation filter that was used at the encoder. Entropy decoding unit 52 may entropy decode the received bitstream to parse out the transformed and quantized residual blocks, which are sent to inverse transform quantization unit 56 and inverse transform unit 58, and to parse out the interpolation syntax element and motion vectors, which are sent to motion compensation unit 55.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 19[b] using said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision; | Karczewicz II, as evidenced by the example citations below, discloses using the first reference block to obtain a first prediction, the first prediction having a second precision, which is higher than the first precision.<br><br>*See, e.g.,* |



Nokia Exhibit 2011, p. 128
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *See limitation 1[b].*<br><br>Karczewicz II, ¶0008:<br><br>> In general, this disclosure describes interpolation techniques performed by an encoder and a decoder during the motion compensation process of video coding. According to the techniques of this disclosure, the encoder may apply a plurality of pre-defined interpolation filters in order to generate a plurality of different interpolated prediction data. The interpolated prediction data that achieves the highest levels of compression may be selected, and the interpolation filter that was used may be coded as syntax and communicated to the decoding device as part of the coded video data. The interpolation filter may be defined once per predictive unit (e.g., once per prediction frame), or may be defined on a block basis (e.g., different filtering may be applied to different video blocks within a predictive unit). Alternatively, the interpolation filter may be defined on a sample basis, or may be defined on a sub-sample basis.<br><br>Karczewicz II, ¶0009:<br><br>> The decoder receives the coded video data, and can interpret the syntax in order to identify the interpolation filter that was used by the encoder. In this way, the decoder can identify and use the same interpolation filters during its motion compensation processes that were used during the encoding process. By considering multiple different interpolation filters at the encoder, compression may be improved relative to techniques that have a fixed interpolation filter. At the same time, the techniques may be significantly less complex than conventional adaptive interpolation filtering, which adaptively defines the interpolation filter coefficients as part of the coding process. The different interpolation filters, according to some aspects of this disclosure, may be pre-defined by implementing adaptive interpolation filtering or other techniques on test video sequences in order to pre-define sets of interpolation filter |



Nokia Exhibit 2011, p. 129
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | coefficients that will likely result in good compression. Alternatively, different 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.<br><br>Karczewicz II, ¶0013:<br><br>In another example, this disclosure provides a method of decoding video data. The decoding method comprises receiving a syntax element that identifies an interpolation filter from a plurality of different pre-defined interpolation filters, generating prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element, and decoding the video data based on the prediction data.<br><br>Karczewicz II, ¶0016:<br><br>In another example, this disclosure provides an apparatus that decodes video data, the apparatus comprising a video decoder that includes a motion compensation unit. The video decoder receives a syntax element that identifies an interpolation filter from a plurality of different pre-defined interpolation filters. The motion compensation unit generates prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element. The video decoder then decodes the video data based on the prediction data.<br><br>Karczewicz II, ¶0041:<br><br>In the example of FIG. 1, source device 12 may include a video source 20, video encoder 22, a modulator/demodulator (modem) 23 and a transmitter 24. Destination device 16 may include a receiver 26 a modem 27, a video decoder 28, and a display device 30. In accordance |



Nokia Exhibit 2011, p. 130
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | with this disclosure, video encoder 22 of source device 12 may be configured to apply one or more of the interpolation techniques of this disclosure as part of a video encoding process. Similarly, video decoder 28 of destination device 16 may be configured to apply one or more of the interpolation techniques of this disclosure as part of a video decoding process.<br><br>Karczewicz II, ¶0042:<br><br>The illustrated system 10 of FIG. 1 is merely exemplary. The interpolation techniques of this disclosure may be performed by any encoding device that supports motion compensated interpolation to sub-pixel resolution. Source device 12 and destination device 16 are merely examples of such coding devices. In this case, source device 12 generates coded video data for transmission to destination device 16. Devices 12, 16 may operate in a substantially symmetrical manner such that each of devices 12, 16 include video encoding and decoding components, e.g., in a combined encoder-decoder (CODEC). Hence, system 10 may support one-way or two-way video transmission between video devices 12, 16, e.g., for video streaming, video playback, video broadcasting, or video telephony.<br><br>Karczewicz II, ¶0046:<br><br>Video encoder 22 and video decoder 28 may operate according to a video compression standard, such as the ITU-T H.264 standard, alternatively described as MPEG-4, Part 10, Advanced Video Coding (AVC). The techniques of this disclosure, however, are not limited to any particular coding standard. Although not shown in FIG. 1, in some aspects, video encoder 22 and video decoder 28 may each be integrated with an audio encoder and decoder, and may include appropriate MUX-DEMUX units, or other hardware and software, to handle encoding of both audio and video in a common data stream or separate data streams. If applicable, MUX-DEMUX units may conform to the ITU H.223 multiplexer protocol, or other protocols such as the user datagram protocol (UDP). |



Nokia Exhibit 2011, p. 131
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0063:<br><br>> Video encoder 22 and video decoder 28 may operate according to a video compression standard, such as the ITU-T H.264 standard, alternatively described as MPEG-4, Part 10, Advanced Video Coding (AVC). The techniques of this disclosure, however, are not limited to any particular coding standard. Although not shown in FIG. 1, in some aspects, video encoder 22 and video decoder 28 may each be integrated with an audio encoder and decoder, and may include appropriate MUX-DEMUX units, or other hardware and software, to handle encoding of both audio and video in a common data stream or separate data streams. If applicable, MUX-DEMUX units may conform to the ITU H.223 multiplexer protocol, or other protocols such as the user datagram protocol (UDP).<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 19[c] using said second reference block to obtain a second prediction, said second prediction having the second precision; | Karczewicz II, as evidenced by the example citations below, discloses using the second reference block to obtain a second prediction, the second prediction having the second precision.<br><br>*See, e.g.,*<br><br>*See limitation 1[c].*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9.* |



Nokia Exhibit 2011, p. 132
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 19[d] obtaining a combined prediction based at least partly upon said first prediction and said second prediction; | Karczewicz II, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction.<br><br>*See, e.g.,*<br><br>*See limitation 1[d].*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 19[e] decreasing a precision of said combined prediction by shifting bits of the combined prediction to the right; and | Karczewicz II, as evidenced by the example citations below, discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right.<br><br>*See, e.g.,*<br><br>*See limitation 1[e].*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 19[f] reconstructing the block of pixels based on the combined prediction. | Karczewicz II, as evidenced by the example citations below, discloses reconstructing the block of pixels based on the combined prediction.<br><br>*See, e.g.,* |

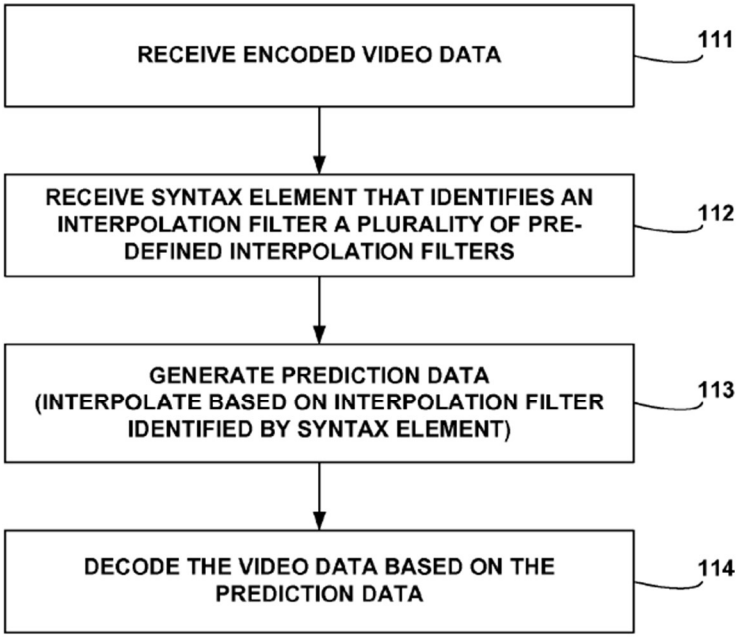


Nokia Exhibit 2011, p. 133
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0111:<br><br>> Motion compensation unit 55 generates prediction data (113), which includes interpolations based on the interpolation filter identified by the syntax element. Video decoder 60 decodes the video data based on the prediction data (114). In particular, the prediction data generated by motion compensation unit 55 may be combined with residual data via adder 64 to reconstruct the video data in the pixel domain. Decoded video may then be stored in reference frame store 62 for subsequent use in prediction, and may also be output for presentation to the user, e.g., via a display or other output device.<br><br>Karczewicz II, Fig. 6: |



Nokia Exhibit 2011, p. 134
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  <br>FIG. 6 |

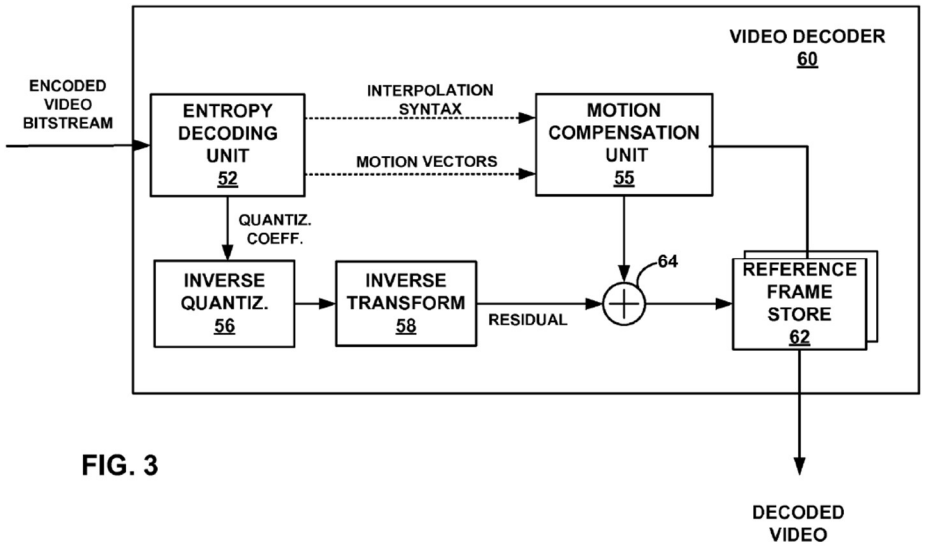


Nokia Exhibit 2011, p. 135
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE:<br>KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0065:<br><br>> Summer 64 decodes residual blocks by summing the residual blocks with the corresponding prediction blocks generated by motion compensation unit 55 to form decoded blocks. If desired, a deblocking filter may also be applied to filter the decoded blocks in order to remove blockiness artifacts. The decoded video blocks are then stored in reference frame store 62, which is a storage element that provides reference blocks for subsequent motion compensation and also produces decoded video to a drive display device (such as device 30 of FIG. 1).<br><br>Karczewicz II, Fig. 3: |



Nokia Exhibit 2011, p. 136
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| |  <br>FIG. 3<br><br>Karczewicz II, ¶0006:<br><br>A coded video block may be represented by prediction information that can be used to create or identify a predictive block, and a residual block of data indicative of differences between the block being coded and the predictive block. The prediction information may comprise the one or more motion vectors that are used to identify the predictive block of data. Given the motion vectors, the decoder is able to reconstruct the predictive blocks that were used to code the residual. Thus, given a set of residual blocks and a set of motion vectors (and possibly some additional syntax), the decoder may be able to reconstruct a video frame that was originally encoded. Inter-coding based on motion estimation and motion compensation can |



Nokia Exhibit 2011, p. 137
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | achieve very good compression because successive video frames or other types of coded units are often very similar. An encoded video sequence may comprise blocks of residual data, motion vectors, and possibly other types of syntax.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the first motion vector points to a subpixel, the first prediction is obtained by interpolation using pixel values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 2.*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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 | Karczewicz II, as evidenced by the example citations below, discloses that the first prediction is obtained by interpolation using values of the first reference block by right shifting a sum of a P-tap filter using values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 3.* |



Nokia Exhibit 2011, p. 138
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| reference block. | *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the second motion vector points to an integer sample, the second prediction is obtained by shifting values of the second reference block to the left.<br><br>*See, e.g.,*<br><br>*See claim 4.*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the decreasing the precision of the combined prediction by shifting bits of the combined prediction to the right, further comprises inserting a rounding offset to the combined prediction before the decreasing.<br><br>*See, e.g.,*<br><br>*See claim 5.*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the* |



Nokia Exhibit 2011, p. 139
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that 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 the first prediction and values of the second prediction.<br><br>*See, e.g.,*<br><br>*See claim 6.*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 25[preamble] An apparatus for decoding a block of pixels, the apparatus comprising: | Karczewicz II, as evidenced by the example citations below, discloses an apparatus for decoding a block of pixels.<br><br>*See, e.g.,*<br><br>*See limitation 19[preamble].*<br><br>Karczewicz II, ¶0003:<br><br>> Digital video capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, digital cameras, digital recording devices, video gaming devices, video game consoles, cellular or satellite radio telephones, and the |



Nokia Exhibit 2011, p. 140
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | like. Digital video devices implement video compression techniques, such as MPEG-2, MPEG-4, or ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), to transmit and receive digital video information more efficiently. Video compression techniques may perform spatial prediction and/or temporal prediction to reduce or remove redundancy inherent in video sequences.<br><br>Karczewicz II, ¶0016:<br><br>In another example, this disclosure provides an apparatus that decodes video data, the apparatus comprising a video decoder that includes a motion compensation unit. The video decoder receives a syntax element that identifies an interpolation filter from a plurality of different pre-defined interpolation filters. The motion compensation unit generates prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element. The video decoder then decodes the video data based on the prediction data.<br><br>Karczewicz II, ¶0017:<br><br>In another example, this disclosure provides an apparatus that interpolates predictive video data for video coding, wherein the apparatus includes a motion compensation unit that generates half-pixel values based on integer pixel values, rounds the half-pixel values to generate half-pixel interpolated values, stores the half-pixel values as non-rounded versions of the half-pixel values, and generates quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0022:<br><br>In another example, this disclosure provides a device that decodes video data, the device |



Nokia Exhibit 2011, p. 141
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | comprising a wireless receiver that receives the video data, and a video decoder including a motion compensation unit that receives a syntax element that identifies an interpolation filter from a plurality of different pre-defined interpolation filters, generates prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element, and decodes the video data based on the prediction data.<br><br>Karczewicz II, ¶0024:<br><br>In another example, this disclosure provides a device that decodes video data, the device comprising a wireless receiver that receives the video data, and a video decoder including a motion compensation unit that decodes the video data. In decoding the video data, the motion compensation unit generates half-pixel values based on integer pixel values, rounds the half-pixel values to generate half-pixel interpolated values, stores the half-pixel values as non-rounded versions of the half-pixel values, and generates quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0025:<br><br>The techniques described in this disclosure may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the software may be executed in one or more processors, such as a microprocessor, application specific integrated circuit (ASIC), field programmable gate array (FPGA), or digital signal processor (DSP). The software that executes the techniques may be initially stored in a computer-readable medium and loaded and executed in the processor. |



Nokia Exhibit 2011, p. 142
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Karczewicz II, ¶0026:<br><br>> Accordingly, this disclosure also contemplates a computer-readable storage medium comprising instructions that upon execution cause a device to encode video data, wherein the instructions cause the device to generate prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on a plurality of different pre-defined interpolation filters, and encode the video data based on the prediction data.<br><br>Karczewicz II, ¶0048:<br><br>> Video encoder 22 and video decoder 28 each may be implemented as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic, software, hardware, firmware or any combinations thereof. Each of video encoder 22 and video decoder 28 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined CODEC in a respective mobile device, subscriber device, broadcast device, server, or the like.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 25[a] at least one processor and at least one memory including computer program code, the at least one memory | Karczewicz II, as evidenced by the example citations below, discloses that at least one processor and at least one memory including computer program code, the at least one memory and computer program code. |

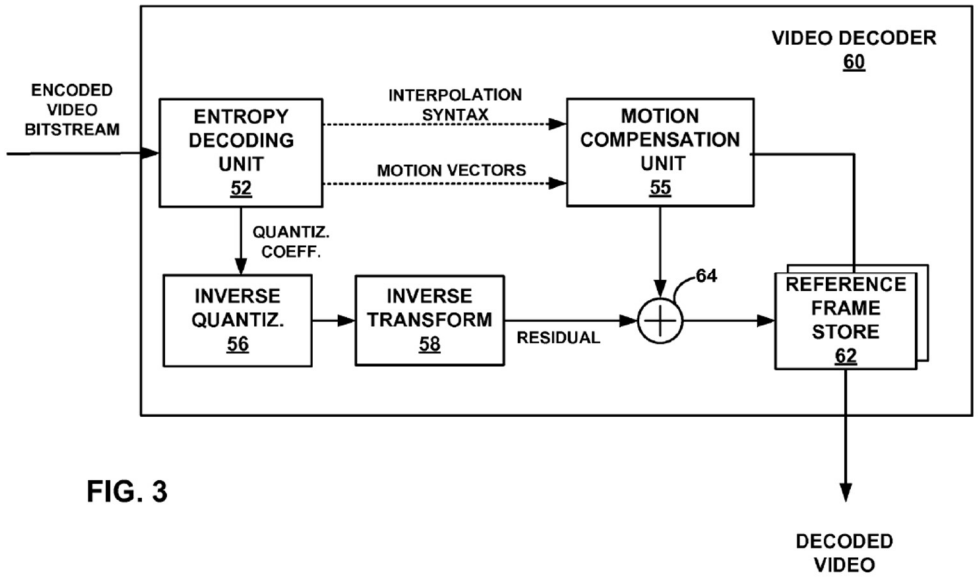


Nokia Exhibit 2011, p. 143
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| and computer program code configured to, with the at least one processor, cause the apparatus to: | *See, e.g.,*<br><br>*See limitation 7[a].*<br><br>Karczewicz II, ¶0006:<br><br>> A coded video block may be represented by prediction information that can be used to create or identify a predictive block, and a residual block of data indicative of differences between the block being coded and the predictive block. The prediction information may comprise the one or more motion vectors that are used to identify the predictive block of data. Given the motion vectors, the decoder is able to reconstruct the predictive blocks that were used to code the residual. Thus, given a set of residual blocks and a set of motion vectors (and possibly some additional syntax), the decoder may be able to reconstruct a video frame that was originally encoded. Inter-coding based on motion estimation and motion compensation can achieve very good compression because successive video frames or other types of coded units are often very similar. An encoded video sequence may comprise blocks of residual data, motion vectors, and possibly other types of syntax.<br><br>Karczewicz II, ¶0063:<br><br>> FIG. 3 is a block diagram illustrating an example of a video decoder 60, which decodes a video sequence that is encoded in the manner described herein. Video decoder 60 includes a motion compensation unit 55 that performs the interpolation techniques of this disclosure for decoding. In particular, on the decoding side, motion compensation unit 55 may receive a syntax element from entropy decoding unit 52 that identifies an interpolation filter from a plurality of different pre-defined interpolation filters. Motion compensation unit 55 may generate prediction data, which includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element. Specifically, motion |



Nokia Exhibit 2011, p. 144
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | compensation unit 55 may generate the prediction data based on motion vectors received from entropy decoding unit 52 and the interpolations as defined by syntax element (labeled interpolation syntax in FIG. 3). Based on this interpolated prediction data, the video data (e.g., a reconstructed residual video block) can be decoded.<br><br>Karczewicz II, Fig. 3:<br><br>  <br><br>FIG. 3<br><br>Karczewicz II, ¶0064: |



Nokia Exhibit 2011, p. 145
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | Entropy decoding unit 52 entropy decodes the received bitstream to generate quantized coefficients and the syntax (e.g., interpolation syntax and motion vectors sent to motion compensation unit 55). Inverse quantization unit 56 inverse quantizes, i.e., de-quantizes, the quantized block coefficients. The inverse quantization process may be a conventional process as defined by H.264 decoding. Inverse transform unit 58 applies an inverse transform, e.g., an inverse DCT or conceptually similar inverse transform process, to the transform coefficients in order to produce residual blocks in the pixel domain. Motion compensation unit 55 produces motion compensated blocks in the manner described herein, e.g., including interpolation based on a set of interpolation filter coefficients identified by the syntax element (i.e., the interpolation syntax).<br><br>Karczewicz II, ¶0110:<br><br>FIG. 6 is a flow diagram consistent with one or more aspects of video decoding consistent with this disclosure. FIG. 6 will be discussed from the perspective of video encoder 50 shown in FIG. 2. In particular, video encoder 60 receives an encoded video data (111), and receives a syntax element that identifies an interpolation filter from a plurality of interpolation filters (112). The encoded bitstream may include both the encoded residual video data and the syntax element that identifies the interpolation filter that was used at the encoder. Entropy decoding unit 52 may entropy decode the received bitstream to parse out the transformed and quantized residual blocks, which are sent to inverse transform quantization unit 56 and inverse transform unit 58, and to parse out the interpolation syntax element and motion vectors, which are sent to motion compensation unit 55.<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the* |



Nokia Exhibit 2011, p. 146
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| | *references identified in Exhibit E-9* |
| 25[b] 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; | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,*<br><br>*See limitation 19[a].* |
| 25[c] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision; | Karczewicz II, as evidenced by the example citations below, discloses using the first reference block to obtain a first prediction, the first prediction having a second precision, which is higher than the first precision.<br>*See, e.g.*,<br><br>*See limitation 19[b].* |
| 25[d] use said second reference block to obtain a second prediction, said second prediction having the second precision; | Karczewicz II, as evidenced by the example citations below, discloses using the second reference block to obtain a second prediction, the second prediction having the second precision.<br><br>*See, e.g.,*<br><br>*See limitation 19[c].* |



Nokia Exhibit 2011, p. 147
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 25[e] obtain a combined prediction based at least partly upon said first prediction and said second prediction; | Karczewicz II, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction.<br><br>*See, e.g.,*<br><br>*See limitation 19[d].* |
| 25[f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and | Karczewicz II, as evidenced by the example citations below, discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right.<br><br>*See, e.g.,*<br><br>*See limitation 19[e].* |
| 25[g] reconstruct the block of pixels based on the combined prediction. | Karczewicz II, as evidenced by the example citations below, discloses reconstructing the block of pixels based on the combined prediction.<br><br>*See, e.g.,*<br><br>*See limitation 19[f].* |
| 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 | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the first motion vector points to a subpixel, the first prediction is obtained by interpolation using pixel values of the first reference block.<br><br>*See, e.g.,* |



**Nokia Exhibit 2011, p. 148**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| using pixel values of said first reference block. | *See claim 20.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the first prediction is obtained by interpolation using values of the first reference block by right shifting a sum of a P-tap filter using values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 21.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the second motion vector points to an integer sample, the second prediction is obtained by shifting values of the second reference block to the left.<br><br>*See, e.g.,*<br><br>*See claim 22.* |
| 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 | Karczewicz II, as evidenced by the example citations below, discloses that the at least one memory and computer code are configured to cause the apparatus to decrease the precision of the combined prediction by shifting bits of the combined prediction to the right, by inserting a rounding offset to the combined prediction before the decreasing.<br><br>*See, e.g.,* |



**Nokia Exhibit 2011, p. 149**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| shifting bits of the combined prediction to the right, by: inserting a rounding offset to the combined prediction before said decreasing. | *See claim 23.*<br><br>*Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz II, in combination with the knowledge of a person of ordinary skill in the art, or in light of one or more of any of the references identified in Exhibit E-9* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that 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 the first prediction and values of the second prediction.<br><br>*See, e.g.,*<br><br>*See claim 24.* |
| 31[preamble] 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 | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,*<br><br>*See limitations 19[preamble], 25[preamble], and 25[a].* |



Nokia Exhibit 2011, p. 150
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| comprising program code instructions configured to: | Karczewicz II, ¶0028:<br><br>In addition, this disclosure also contemplates a computer-readable storage medium comprising instructions that upon execution cause a device to interpolate predictive video data for video coding, wherein the instructions cause the device to generate half-pixel values based on integer pixel values, round the half-pixel values to generate half-pixel interpolated values, store the half-pixel values as non-rounded versions of the half-pixel values, and generate quarter-pixel values based on the non-rounded versions of the half-pixel values and the integer pixel values.<br><br>Karczewicz II, ¶0027:<br><br>This disclosure also contemplates a computer-readable storage medium comprising instructions that upon execution cause a device to decode video data, wherein the instructions cause the device to, upon receiving a syntax element that identifies an interpolation filter from a plurality of different pre-defined interpolation filters, generate prediction data, wherein generating the prediction data includes interpolating pixel values of reference video data based on the interpolation filter identified by the syntax element, and decode the video data based on the prediction data. |
| 31[a] 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 | Karczewicz II, as evidenced by the example citations below, discloses 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.<br><br>*See, e.g.,* |



Nokia Exhibit 2011, p. 151
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| second reference block have values with a first precision; | *See limitation 19[a].* |
| 31[b] use said first reference block to obtain a first prediction, said first prediction having a second precision, which is higher than said first precision; | Karczewicz II, as evidenced by the example citations below, discloses using the first reference block to obtain a first prediction, the first prediction having a second precision, which is higher than the first precision.<br><br>*See, e.g.,*<br><br>*See limitation 19[b].* |
| 31[c] use said second reference block to obtain a second prediction, said second prediction having the second precision; | Karczewicz II, as evidenced by the example citations below, discloses using the second reference block to obtain a second prediction, the second prediction having the second precision.<br><br>*See, e.g.,*<br><br>*See limitation 19[c].* |



Nokia Exhibit 2011, p. 152
Snap/Hisense v. Nokia
IPR2025-01375

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 31[d] obtain a combined prediction based at least partly upon said first prediction and said second prediction; | Karczewicz II, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction.<br><br>*See, e.g.,*<br><br>*See limitation 19[d].* |
| 31[e] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and | Karczewicz II, as evidenced by the example citations below, discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right.<br><br>*See, e.g.,*<br><br>*See limitation 19[e].* |
| 31[f] reconstruct the block of pixels based on the combined prediction. | Karczewicz II, as evidenced by the example citations below, discloses reconstructing the block of pixels based on the combined prediction.<br><br>*See, e.g.,*<br><br>*See limitation 19[f].* |
| 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 | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the first motion vector points to a subpixel, the first prediction is obtained by interpolation using pixel values of the first reference block.<br><br>*See, e.g.,* |



**Nokia Exhibit 2011, p. 153**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| interpolation using pixel values of said first reference block. | *See claim 20.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that the first prediction is obtained by interpolation using values of the first reference block by right shifting a sum of a P-tap filter using values of the first reference block.<br><br>*See, e.g.,*<br><br>*See claim 21.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that in an instance in which the second motion vector points to an integer sample, the second prediction is obtained by shifting values of the second reference block to the left.<br><br>*See, e.g.,*<br><br>*See claim 22.* |
| 35. The computer program product according to claim 31, wherein the program code instructions configured to decrease said precision of said combined prediction by | Karczewicz II, as evidenced by the example citations below, discloses that the program code instructions configured to decrease the precision of the 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 the decreasing. |



**Nokia Exhibit 2011, p. 154**
**Snap/Hisense v. Nokia**
**IPR2025-01375**

| ASSERTED CLAIMS | PRIOR ART REFERENCE: KARCZEWICZ II |
|---|---|
| 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. | *See, e.g.,*<br><br>*See claims 23 and 29.* |
| 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. | Karczewicz II, as evidenced by the example citations below, discloses that 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 the first prediction and values of the second prediction.<br><br>*See, e.g.,*<br><br>*See claim 24.* |



**Nokia Exhibit 2011, p. 155**
**Snap/Hisense v. Nokia**
**IPR2025-01375**