EXHIBIT E-6

KARCZEWICZ I AND KARCZEWICZ II

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 2011/0007799 ("Karczewicz I"), entitled "Non-Zero Rounding and Prediction Mode Selection Techniques in Video Encoding," published on January 13, 2011. Karczewicz I was filed July 9, 2009, and is prior art under at least 35 U.S.C. § 102 (e). Claims 1-36 of the '267 patent are invalid in view of US Patent Application Publication 2009/0257499 ("Karczewicz II"), 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).

Karczewicz I and Karczewicz II are analogous art in the same field as the '267 patent because they are directed to video encoding/decoding, motion prediction, and H.264. Karczewicz I teaches block-based techniques for motion prediction, including bi-directional predictions calculated by averaging predictions based on two reference blocks, which may be calculated using sub-pixel interpolation, as set forth in further detail below. *See* Karczewicz I, ¶¶35, 41-44, 55-60. Karczewicz II reduced rounding inaccuracies with interpolated pixels by maintaining higher precision for intermediate values while delaying rounding until later in the process. *See* Karczewicz II, ¶¶10, 39, 53, 59, 93-106. Karczewicz II further discloses generating half-pixel values, storing the half-pixel values as non-rounded versions and combining them based on the non-rounded versions of the half-pixel values and the integer pixel values. *See* Karczewicz II, ¶¶17, 96-108.

A PHOSITA would have been motivated to combine Karczewicz II's interpolation disclosures and teachings with the bi-prediction models and related disclosures of Karczewicz I for numerous reasons.

First, Karczewicz I and Karczewicz II's similarities would have motivated a POSITA to implement teachings from Karczewicz I and Karczewicz II using their common architecture, combining prior art elements according to known methods. Karczewicz I and Karczewicz II are Qualcomm patent applications by the same inventors. Both apply their teachings to similar video-coding architectures. *See* Karczewicz I, ¶¶2, 29-50 and FIG. 1; Karczewicz II, ¶¶2, 40-53 and FIG. 1. Both are directed to block-based H.264 motion prediction. *See* Karczewicz I, ¶¶35, 41-44, 55-60; Karczewicz II, ¶¶8, 35-36, 46, 54. Both teach a video decoder that performs

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.

motion compensation for block-based decoding. *See* Karczewicz I, ¶89 and FIG. 4; Karczewicz II, Fig. 3 and ¶ 63. Karczewicz I teaches bi-prediction techniques for H.264 with two motion vectors pointing to two blocks of pixels that are averaged together. *See* Karczewicz I, ¶¶35, 44, 60. Beyond integer pixels, Karczewicz II teaches that H.264 motion vectors can also point to fractional sub-pixels (*See* Karczewicz II at paragraphs 56-58, 93-102) and Karczewicz II teaches improved calculations for averaging interpolated pixel values, where rounding is delayed until later in the process, thereby maintaining higher precision for intermediate calculations. *See* Karczewicz II, ¶¶10, 20-24, 39, 93-106. Therefore, Karczewicz II provides complementary teachings that improve Karczewicz I by reducing rounding inaccuracies. *See* Karczewicz II, ¶¶39, 53, 59 and 102.

Second, a PHOSITA would have been motivated to use Karczewicz II's known techniques to improve similar devices/methods, as taught by Karczewicz I, in the same way to improve prediction accuracy. Karczewicz I and Karczewicz II both encompass motion prediction calculations that average interpolated pixel values. Karczewicz II teaches an improved calculation for averaging two half-pixel values in bi-directional prediction. *See* Karczewicz II, ¶¶101, 103, 163-166; FIG. 4D. The calculations taught by Karczewicz II correspond to the calculations used for Karczewicz I's bi-predicted pixel values when applied to integer and sub-pixel values. And it would have had predictable results because it applies teachings from Karczewicz II to corresponding mathematical calculations used for Karczewicz I, using similar video codec architectures.

Third, a PHOSITA would have recognized the applicability of Karczewicz II to the corresponding calculations in Karczewicz I, which was a simple matter given the level of ordinary skill. Notably, a POSITA would have understood fundamental computer logic concepts (e.g., binary arithmetic, bit shifting, precision control, rounding and offsetting, and error analysis) and mathematical calculations for known motion estimation and compensation techniques (e.g., bi-directional prediction, determination and use of motion vectors, interpolation for fractional pixels) because they are integral to working with video codecs. Karczewicz I teaches that the inter-predictive coding process includes interpolation used in H.264, providing express teaching, suggestion, and motivation ("TSM") to combine with known interpolation teachings. *See* Karczewicz I, ¶41. Karczewicz II teaches optimizations for interpolating and averaging half pixels which may be applied during inter-coding including bi-directional prediction in accordance with H.264. *See* Karczewicz II, ¶¶51,54, 56-58, 60, 93-102.

Fourth, the combination would not have changed the principle of operation of either reference, but merely includes the use Karczewicz II's known techniques to improve Karczewicz I's similar devices or methods in the same way. *See* Karczewicz I, ¶¶29-50, 53, FIG. 1 and FIG. 2; Karczewicz II, ¶¶40-54, 56, FIG. 1 and FIG.2. Given the similarities between Karczewicz I and Karczewicz II, a POSITA would have understood that Karczewicz II's techniques are readily applicable to Karczewicz I. Moreover, the combination merely

Nokia Exhibit 2006, p. 2
ASUS v. Nokia
IPR2025-01154

changes when rounding occurs for calculations that already included rounding. *See* Karczewicz I, ¶60 and FIG. 2; Karczewicz II, ¶¶96-106. This minor implementation detail would not have changed the principle of operation of Karczewicz I or Karczewicz II.

Fifth, a PHOSITA would have had a reasonable expectation of success because the combination applies the math behind Karczewicz II's improved calculations to Karczewicz I's bi-directional prediction without further modification to reduce a rounding error that occurs in bi-directional prediction. Consistent with this, Karczewicz II that its disclosed interpolation techniques are applicable to bi-prediction (Karczewicz II, ¶54), and further discloses that "keep[ing] the highest possible precision through the intermediate steps" and avoiding "any shifting, rounding and clipping operations[] until the very last step of the interpolation process." *See* Karczewicz II, ¶¶53, 59, 99, 102. Furthermore, a POSITA would have been more than capable of applying Karczewicz II's teachings because Karczewicz II's calculations involve basic mathematic and logical operations (e.g., addition and bit-shifting) that were well known or taught in high-school or undergraduate-level courses, and basic video codec operations that were a core part of industry work in video codecs.

The Combination of Karczewicz I and Karczewicz II

H.264 includes interpolated fractional pixel positions. Karczewicz II explains that H.264 calculates predictions for half-pixel positions using 6-tap filters that interpolate the sub-pixel based on nearby pixels in the same row (e.g., half-pixel "b") or column (e.g., half-pixel "h"). Karczewicz II, ¶74, ¶¶93-94, Fig. 4A-4B:

Nokia Exhibit 2006, p. 3
ASUS v. Nokia
IPR2025-01154

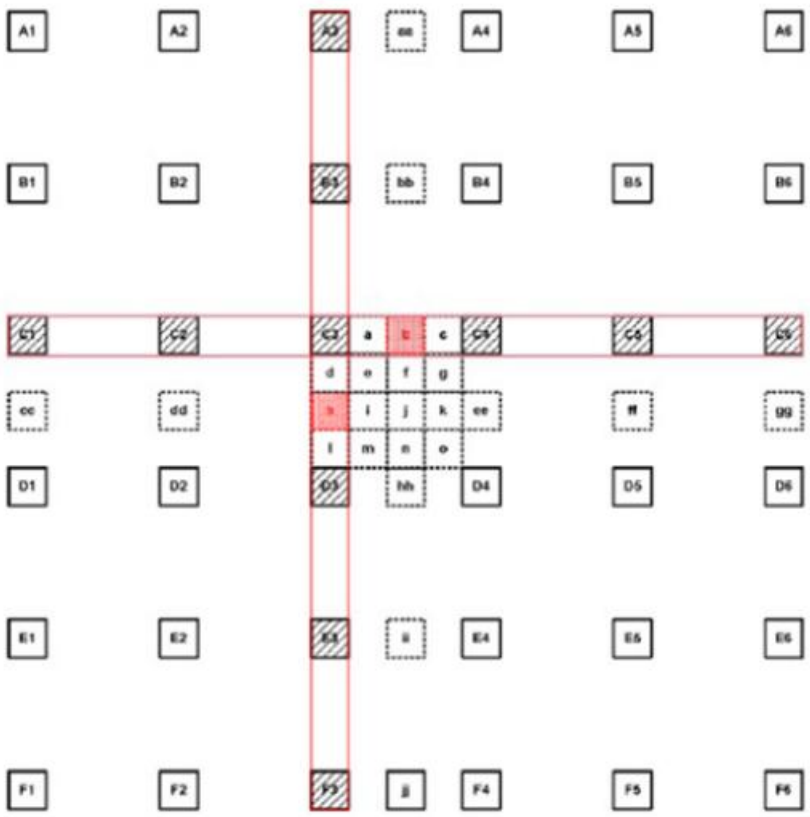


FIG. 4B

This process involved two steps. First, the filter multiplied the six pixels in the row (or column) by filter values and added the products together to produce a non-rounded prediction (e.g., b1):

$$b1=C1-5*C2+20*C3+20*C4-5*C5+C6$$

Second, the result was rounded, e.g., right-shifted (“>>”) producing rounded prediction b:

Nokia Exhibit 2006, p. 4
ASUS v. Nokia
IPR2025-01154

b=max(0, min(255, (b1+16)>>5))

Karczewicz II, ¶¶93-4, Fig. 4B.

For half-pixel positions in the center of four integer pixels (e.g., j), interpolation was applied in two rounds: first to interpolate a middle row of half-pixels, second to interpolate that middle row into the center-pixel, applying the using the same 6-tap interpolation filter and two-step process described above to the interpolated middle row. Karczewicz II, ¶95, Fig. 4C.

For quarter-pixel predictions, Karczewicz II averaged the two nearest integer or half-pixel predictions as explained above. Karczewicz II, ¶¶96-97. Karczewicz II teaches an improvement by "keep[ing] the highest possible precision through the intermediate steps" and avoiding "any shifting, rounding and clipping operations[] until the very last step of the interpolation process." Karczewicz II, ¶99, ¶102, ¶10, ¶39, ¶53, ¶59.

The combination of Karczewicz I and Karczewicz II teach three different calculation scenarios that render the claims of the '267 patent obvious. In particular, Karczewicz I teaches weighted bi-directional prediction that averages two predictions based on two reference blocks (e.g., pred0(i,j), pred1(i,j)) with an offset (e.g., +1), including a default mode (Karczewicz I, ¶55) with equal weights that performs a simple average. Karczewicz I, ¶60:

pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1

Karczewicz I, ¶¶58-60. Because H.264 allows motion vectors to point to integer pixels or fractional pixels/subpixels (Karczewicz I, ¶41; Karczewicz II, ¶¶93-102, ¶¶56-58), the predictions pred0(i,j) and pred1(i,j) encompass scenarios that include integer pixel prediction, a half-pixel prediction, or a center-pixel prediction, and Karczewicz-I calculates an average of these pixel values.

Consistent with H.264, Karczewicz I teaches that the inter-predictive coding process includes interpolation, providing express teaching, suggestion, and motivation to combine with known interpolation teachings. Karczewicz I, ¶41. Karczewicz II teaches optimizations for interpolating and averaging integer, half-, and center-pixels, which a POSITA would have been motivated to apply to at least three scenarios for Karczewicz I's teachings.

Scenario 1 (with the first motion vector pointing to a half-pixel position and the second motion vector pointing to an integer pixel position). Beyond integer pixels, Karczewicz II explains how, for H.264, motion vectors can also point to half-pixel positions (Karczewicz II, ¶¶93-102, ¶¶56-58), which therefore teaches or suggests the default weighted prediction calculated as an average of a half-pixel and integer pixel. *See* Karczewicz I, ¶60, ¶55; Ex-1003, ¶151.

Nokia Exhibit 2006, p. 5
ASUS v. Nokia
IPR2025-01154

Karczewicz II teaches an improved calculation for averaging a half-pixel value with an integer pixel value. *See* Karczewicz I, ¶96, Fig. 4D. Karczewicz II first explains the conventional calculation for averaging two numbers. Karczewicz I, Fig. 4D, Table 1:

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 improves this conventional approach by replacing the equations in Table 1 with those in Table 3, where the pixel values are combined at a higher precision. Karczewicz II, 99. The integer pixel value (e.g., C3) is multiplied by 32 (left-shifted 5 bits), taking its precision from 8 to 13 bits. Karczewicz II, Table 5. Instead of using a rounded 8-bit half-pixel (e.g., b), KarczewiczII delays the rounding step and instead uses a non-rounded half-pixel prediction (e.g., b1) that is 15 bits. *Id.* These pixel values are combined, along with a rounding offset of 32, before rounding to reduce the precision at the end, e.g., shifting 6 bits to the right (">>6"). *Id.*, Table 3:

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

Nokia Exhibit 2006, p. 6
ASUS v. Nokia
IPR2025-01154

The operations for this improved approach are shown in Table 5 of Karczewicz II. Karczewicz II, ¶103, Table 5:

TABLE 5

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

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

Since Karczewicz II teaches this optimization for averaging an interpolated half-pixel and an integer pixel, a POSITA would have been motivated to apply that improved calculation to Karczewicz I's default weighted prediction equation, which likewise calculates an average, with the half-pixel prediction (e.g., pred0(i,j)) kept at a higher, non-rounded precision and the integer pixel prediction (e.g.,

Nokia Exhibit 2006, p. 7
ASUS v. Nokia
IPR2025-01154

pred1(i,j)) left-shifted. *See* Karczewicz I, ¶60, ¶55; Ex-1006, ¶¶93-94, ¶99, Table 3. As Karczewicz II teaches, these values are combined with a rounding offset of 32 and then right-shifted 6 bits to reduce the precision. This combination results in the following equation:

pred(i,j)=(non-rounded(pred0(i,j))+pred1(i,j)<<5+32)>>6

See Karczewicz II, ¶96, ¶99, Tables 1 and 3; Karczewicz I, ¶60.

Scenario 2 (with the first motion vector pointing to a center-pixel position and the second motion vector pointing to a half-pixel position). The default weighted prediction is calculated as an average of a center-pixel and a half-pixel. See Karczewicz I, ¶60, ¶55.

Karczewicz II teaches an improved calculation for averaging a center- and half-pixel. Ex-1006, Fig. 4D. Karczewicz II first explains the conventional calculation for averaging two numbers. Karczewicz II, ¶96, Fig. 4D, Table 1:

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 improves this conventional approach by replacing the equations in Table 1 with those in Table 3, where the pixel values are combined at a higher precision. Karczewicz II, ¶99. Whereas Table 1 used a fully-rounded center-pixel (e.g., "j"), Table 3 uses a partially-rounded pixel (e.g., "j1>>5") that remains 5 bits longer than the rounded value in Table 1.[1] Likewise, Table 3 replaces the rounded half-pixel (e.g., "b") with a non-rounded half-pixel (e.g., "b1") that is 15 bits. These pixel values are combined, along with a

[1] Here, the partially-rounded prediction is obtained by shifting the non-rounded prediction j1 to the right by 5 bits. This is fewer bits than the 10 bits shifted for calculating the fully rounded prediction j. Karczewicz II, ¶95.

Nokia Exhibit 2006, p. 8
ASUS v. Nokia
IPR2025-01154

rounding offset of 32, before rounding to reduce the precision at the end, e.g., shifting 6 bits to the right (“>>6”). Karczewicz II, Table 3:

TABLE 3

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

Nokia Exhibit 2006, p. 9
ASUS v. Nokia
IPR2025-01154

The operations for this improved approach are shown in Table 8 of Karczewicz II. Karczewicz II, ¶105, Table 8:

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
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

Since Karczewicz II teaches this optimization for averaging interpolated center- and half-pixels, a POSITA would have been motivated to apply that improved calculation to Karczewicz I's default weighted prediction equation, which likewise calculates an average, with the center-pixel prediction partially rounded and the half-pixel prediction non-rounded. *See* Karczewicz I, ¶60, ¶55; *e.g.*, Karczewicz II, ¶10, ¶20, ¶¶23-24, ¶39, ¶74, ¶93, ¶¶99-103. This combination results in the following equation:

pred(i,j)=(non-rounded(pred0(i,j))>>5+non-rounded(pred1(i,j)) +32)>>6

See Karczewicz II, ¶96, ¶99, Tables 1 and 3; Karczewicz I, ¶60.

Scenario 3 (with both motion vectors pointing to half-pixel positions). The default weighted prediction is calculated as an average of two half-pixels. *See* Karczewicz I, ¶60, ¶55.

Nokia Exhibit 2006, p. 10
ASUS v. Nokia
IPR2025-01154

Karczewicz II teaches an improved calculation for averaging two half-pixel values. Karczewicz II, Fig. 4D. Karczewicz II explains the conventional calculation for averaging two numbers. Karczewicz II, ¶97, Fig. 4D, Table 2:

TABLE 2

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

Karczewicz II improves this conventional approach replacing the equations in Table 2 with those in Table 4, where the pixel values are combined at a higher precision. Karczewicz II, ¶101. Instead of using a rounded 8-bit half-pixel (e.g., "b" or "h"), Karczewicz II delays the rounding step and instead uses non-rounded half-pixels (e.g., "b1" and "h1") that are 15 bits each. Id. These pixel values are combined, along with a rounding offset of 32, before rounding to reduce the precision at the end, e.g., shifting 6 bits to the right (">>6"). *Id.*, Table 4:

TABLE 4

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

The operations for this improved approach are shown in Table 6 of Karczewicz II. Karczewicz II, ¶103, Table 6:

Nokia Exhibit 2006, p. 11
ASUS v. Nokia
IPR2025-01154

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

Since Karczewicz II teaches this optimization for averaging two interpolated half-pixels, a POSITA would have been motivated to apply that improved calculation to Karczewicz I's default weighted prediction equation, which likewise calculates an average, with the half-pixel predictions non-rounded. *See* Karczewicz I, ¶60, ¶55; Karczewicz II, ¶¶97-101. This combination results in the following equation:

pred(i,j)=(non-rounded(pred0(i,j))+non-rounded(pred1(i,j)) +32)>>6

See Karczewicz II, ¶97, ¶101, Tables 2 and 4; Karczewicz I, ¶60.

In addition to these teachings, it was well-known to use higher-precision intermediate values for video-coding computations. As a result, the prior art provides an express teaching, suggestion, and motivation to apply such teachings from Karczewicz II to related techniques in Karczewicz I to achieve a higher accuracy. *See, e.g.*, U.S. Patent No. 8,594,188 at Abstract.

Nokia Exhibit 2006, p. 12
ASUS v. Nokia
IPR2025-01154

Preserving higher-precision intermediate values was known to benefit motion compensation. U.S. Patent No. 9,344,744 at 7:4-19 ("a prediction and a reconstruction for a block of a picture to be predicted ... in a higher precision."), 16:22-17:36. This approach was known to result in smaller residual information. *Id.* It was also known that preserving higher-precision intermediate values improved the accuracy of interpolation. Srinivasan, An Overview of VC-1, 000005; U.S. Patent Application Publication No. 2003/0112864, ¶¶61-62.

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

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

A POSITA would have had a reasonable expectation of success because the combination applies the math behind Karczewicz II's improved calculations to Karczewicz I's scenarios without further modification. Furthermore, a POSITA would have been more than capable of applying Karczewicz II's teachings because Karczewicz II's calculations involve basic mathematic and logical operations (e.g., addition and bit-shifting) that were well known or taught in high-school or undergraduate-level courses, and basic video codec operations that were a core part of industry work in video codecs.

ASSERTED CLAIMS	**PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]**
1[preamble] A method for encoding a block of pixels, the method comprising:	Karczewicz I, as evidenced by the example citations below, discloses a method for encoding a block of pixels. Karczewicz I, as evidenced by the example citations below, discloses a method for encoding a block of pixels.

[2] Moreover, Respondents incorporate by reference in their entirety the disclosures and arguments specific to Karczewicz I and Karczewicz II contained within the petitions seeking IPR review in IPR2024-00626 and IPR2024-00627, which further show how these references demonstrate that the claims of the '267 patent are obvious.

Nokia Exhibit 2006, p. 13
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See, e.g., Karczewicz I, ¶0002: > The disclosure relates to video encoding and, more particularly, video encoding techniques that use bi-directional prediction. Karczewicz I, ¶0008: > This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block may be predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, techniques are described for selecting among default weighted prediction, implicit weighted prediction, and explicit weighted prediction. In this context, techniques are also described for adding offset to prediction data, e.g., using the format of explicit weighted prediction to allow for offsets to predictive data that is otherwise defined by implicit or default weighted prediction. Karczewicz I, ¶0009: > In one example, this disclosure describes a method of encoding video data, the method comprising selecting between default weighted prediction and implicit weighted prediction, upon selecting default weighted prediction, selecting between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the default weighted prediction, upon selecting implicit weighted prediction, selecting between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the implicit weighted prediction, and encoding the video data using weighed prediction data defined by the selections.

Nokia Exhibit 2006, p. 14
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, ¶0022: This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block is predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, rounding adjustments to bi-directional predictive data may be purposely eliminated to provide predictive data that lacks any rounding bias. In this case, rounded and unrounded predictive data may both be considered in a rate-distortion analysis to identify the best data for prediction of a given video block. One or more syntax elements may be encoded to indicate the selection, and a decoder may interpret the one or more syntax elements in order to determine whether rounding should be used in the decoding process. Karczewicz I, Fig. 1:

Nokia Exhibit 2006, p. 15
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

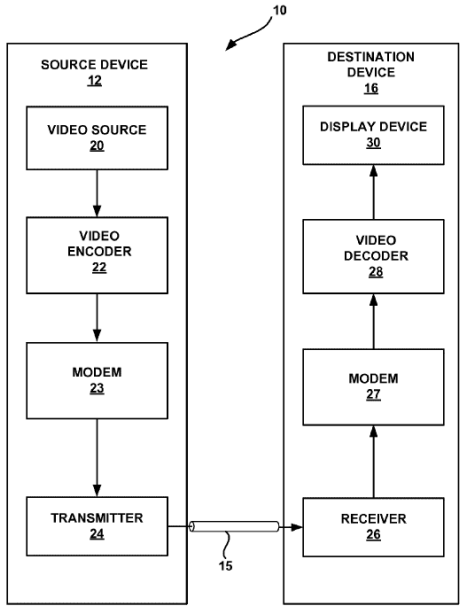


FIG. 1

Karczewicz I, ¶0029-30:

> [0029] FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite

Nokia Exhibit 2006, p. 16
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	radiotelephones, or any devices that can communicate video information over a communication channel 15, which may or may not be wireless. The techniques of this disclosure, however, which concern non-zero rounding and prediction mode selection techniques, are not necessarily limited to wireless applications or settings. [0030] 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 non-zero rounding and prediction mode selection techniques as part of a video encoding process. Video decoder 28 may receive one or more syntax elements indicating the selection and indicating whether non-zero rounding was used. Accordingly, video decoder 28 may perform the proper weighted prediction signaled in the received bistream. Karczewicz I, ¶0032: Video source 20 of source device 12 may include a video capture device, such as a video camera, a video archive containing previously captured video, or a video feed from a video content provider. As a further alternative, video source 20 may generate computer graphics-based data as the source video, or a combination of live video, archived video, and computer-generated video. In some cases, if video source 20 is a video camera, source device 12 and destination device 16 may form so-called camera phones or video phones. In each case, the captured, pre-captured or computer-generated video may be encoded by video encoder 22. The encoded video information may then be modulated by modem 23 according to a communication standard, e.g., such as code division multiple access (CDMA) or another communication standard, and transmitted to destination device 16 via transmitter 24. Modem 23 may include various mixers, filters, amplifiers or other components designed for signal

Nokia Exhibit 2006, p. 17
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

modulation. Transmitter 24 may include circuits designed for transmitting data, including amplifiers, filters, and one or more antennas.

Karczewicz I, Fig. 2:

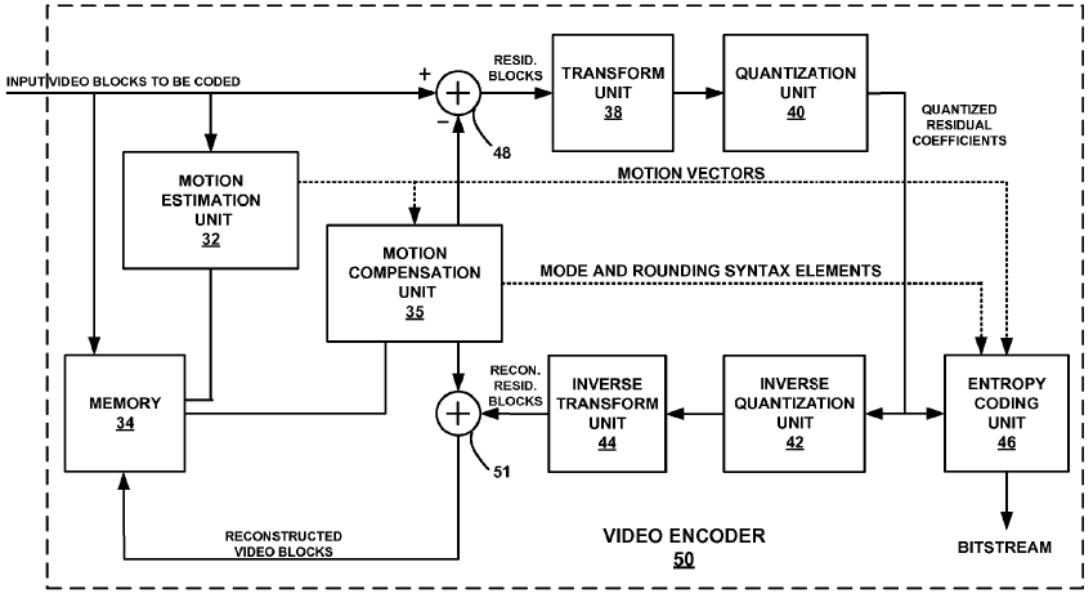


FIG. 2

Karczewicz I, ¶0039:

A video sequence typically includes a series of video frames. Video encoder 22 and video decoder 28 may operate on video blocks within individual video frames in order to encode and

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	decode 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 may include a series of slices or other independently decodable units. Each slice may include a series of macroblocks, which may be arranged into sub-blocks. As an example, the ITU-T 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. Karczewicz I, ¶0040: 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 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 macroblocks and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Alternatively, frames themselves may be decodable units, or other portions of a frame may be defined as decodable units. The term "coded unit" refers to any independently decodable unit of a video frame such as an entire frame, a slice of a frame, a group of pictures (GOPs), or another independently decodable unit defined according to the coding techniques used. Karczewicz I, ¶¶0051-76: [0051] FIG. 2 is a block diagram illustrating an example of a video encoder 50 that may perform techniques consistent with this disclosure. Video encoder 50 may correspond to video encoder 22 of source device 12, or a video encoder of a different device. Video encoder 50 may perform intra- and inter-coding of blocks within video frames, although intra-coding components are

Nokia Exhibit 2006, p. 19
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	not shown in FIG. 2 for ease of illustration. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames of a video sequence. Intra-mode (I-mode) may refer to the spatial based compression mode and Inter-modes such as prediction (P-mode) or bi-directional (B-mode) may refer to the temporal based compression modes. [0052] As shown in FIG. 2, video encoder 50 receives a current video block within a video frame or slice to be encoded. In the example of FIG. 2, video encoder 50 includes motion estimation unit 32, a motion compensation unit 35, a memory 34, an adder 48, a transform unit 38, a quantization unit 40, and an entropy coding unit 46. For video block reconstruction, video encoder 50 also includes an inverse quantization unit 42, an inverse transform unit 44, and an adder 51. Video encoder 50 may also include a deblocking filter (not shown) to filter block boundaries to remove blockiness artifacts from reconstructed video. If desired, the deblocking filter would typically filter the output of adder 51. [0053] 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 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. For demonstrative purposes, the techniques described in this disclosure are described as being performed by motion compensation unit 35.

Nokia Exhibit 2006, p. 20
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0054] Motion estimation unit 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of one or more predictive coded units (e.g., a previous and/or future frame in terms of time or temporally). Motion estimation unit 32 may, as an example, select a motion vector for a B-frame in a number of ways. In one way, motion estimation unit 32 may select a previous or future frame from a first set of frames (referred to as list 0) and determine a motion vector using only this previous or future frame from list 0. Alternatively, motion estimation unit 32 may select a previous or future frame from a second set of frames (referred to as list 1) and determine a motion vector using only this previous or future frame from list 1. In yet another way, motion estimation unit 32 may select a first frame from list 0 and a second frame from list 1 and select one or more motion vectors from the first frame of list 0 and the second frame of list 1. This third form of prediction may be referred to as bi-predictive motion estimation. Techniques of this disclosure may be implemented so as to efficiently select a motion-compensated bi-prediction mode. The selected motion vector for any given list may point to a predictive video block that is most similar to the video block being coded, e.g., as defined by a metric such as sum of absolute difference (SAD) or sum of squared difference (SSD) of pixel values of the predictive block relative to pixel values of the block being coded. [0055] According to the ITU-T H.264/AVC standard, three motion-compensated bi-predictive algorithms or modes may be used to predict a B-frame or portions thereof, such as video blocks, macroblocks or any other discreet and/or contiguous portion of a B-frame. A first motion-compensated bi-predictive algorithm or mode, which is commonly referred to as default weighted prediction, may involve applying default weights to each identified video block of the first frame of list 0 and the second frame of list 1. The default weights may be programmed according to the standard, and are often selected to be equal for default weighted prediction. The weighted blocks of the first and second frames are then added together and divided by the total number of frames used to predict the B-frame, e.g., two in this instance. Often, this division is accomplished by adding 1 to the addition of the weighted blocks of the first and

Nokia Exhibit 2006, p. 21
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	second frames and then shifting the result to the right by one bit. The addition of 1 is a rounding adjustment. [0056] In accordance with one aspect of this disclosure, the addition of 1 (the rounding adjustment) prior to the right shift by one may be avoided, thus eliminating upward biased rounding. Motion compensation unit 35 may generate both a weighted block with rounding and a weighted block without rounding, and may select the block that achieves the best coding efficiency. [0057] More generally, weighted prediction may be given by: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+2^r)>>(r+1)$ where pred(i,j) is data associated with the weighed prediction block, pred0(i,j) is data from list 0, pred1(i,j) is data from list 1, w0 and w1 are the weight factors, 2r is the rounding term, and >> is a right shift operation by (r+1) bits. Consistent with this disclosure, two different version of pred(i,j) may be generated and considered by motion compensation unit 35. The first is consistent with the equation above, and the second is consistent with the equation above without rounding, i.e., with the term "2r" removed from the equation. Eliminating this rounding may achieve better weighted predictive data in some cases, which can improve coding efficiency. Motion compensation unit 35 may generate one or more syntax elements to define whether or not rounding was used for a given video block or set of video blocks. Both the bi-directional prediction mode and the one or more syntax elements indicating whether rounding was used may be output form motion compensation unit 35 to entropy coding unit 46 for inclusion in the encoded bitstream. [0058] B pictures use two lists of previously-coded reference pictures, list 0 and list 1. These two lists can each contain past and/or future coded pictures in temporal order. Blocks in a B picture may be predicted in one of several ways: motion-compensated prediction from a list 0

Nokia Exhibit 2006, p. 22
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	reference picture, motion-compensated prediction from a list 1 reference picture, or motion-compensated prediction from the combination of both list 0 and list 1 reference pictures. To get the combination of both list 0 and list 1 reference pictures, two motion compensated reference areas are obtained from list 0 and list 1 reference picture respectively. Their combination will be used to predict the current block. [0059] In this disclosure, the term "B pictures" will be used to refer generally to any types of B units, which may be B frames, B slices, or possibly other video units that include at least some B video blocks. As mentioned, B pictures may allow 3 types of weighted prediction. For simplicity, only forward prediction in unidirectional prediction is shown below, although backwards prediction could also be used. [0060] Default weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: pred(i,j)=pred0(i,j) Bidirectional prediction: pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1 where pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1. [0061] Implicit weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: pred(i,j)=pred0(i,j) Bidirectional prediction: pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+32)>>6

Nokia Exhibit 2006, p. 23
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	In this case, each prediction is scaled by a weighting factor w0 or w1, where w0 and w1 are calculated based on the relative temporal position of the list 0 and list 1 reference pictures. [0062] Explicit weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: $pred(i,j)=(pred0(i,j)*w0+2^{r-1})>>r+o1$ Bidirectional prediction: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+2^{r})>>(r+1)+((o1+o2+1)>>1)$ In this case, the weighting factors are determined by the encoder and transmitted in the slice header, and o1 and o2 are picture offsets for list 0 and list 1 reference pictures respectively. [0063] Conventionally, rounding adjustments are always used in bidirectional prediction. According to the equations above, a rounding adjustment of 1 is used in the default weighted prediction prior to a right shift by one, and a rounding adjustment of 32 is used in the implicit weighted prediction prior to a right shift by six. Generally, a rounding adjustment of 2^{r-1} is commonly used prior to a right shift by r, where r represents a positive integer. [0064] Such frequent and biased rounding operations can reduce precision of prediction. Furthermore, in the bidirectional prediction of the explicit weighted prediction, there are actually 2 roundings, one for the reference pictures and the other for the offsets. Therefore, in this case, the rounding error can accumulate. In accordance with one aspect of this disclosure, instead of doing 2 separate roundings, the video encoder can add the offsets to the weighted prediction before right shift as the following: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+((o1+o2)<<r)+2^{r})>>(r+1)$, where pred(i,j) is the weighted prediction data associated with rounding, pred0(i,j) is data from list 0, pred1(i,j) is data from list 1, w0 and w1 are weight factors, o1 and o2 are offsets, and r

Nokia Exhibit 2006, p. 24
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	and 2^r are rounding terms that provide the rounding in conjunction with a right shift operation ">>" by (r+1) bits. This may provide better prediction accuracy. In this case, a new syntax element may also be defined to allow for combination of two different offsets (o1 and o2) into one offset. Furthermore, in this case, a rounding value may comprise the rounding adjustment described above (e.g., 2^r prior to a right shift operation by (r+1) bits) as well as another a rounding value ("r") associated with the offset. The equation above may also be modified slightly to provide higher precision for the offsets. If higher precision for the offsets are desired, the offsets may be multiplied by factors (such as by 2) and then rounded to an integer numbers. The left shift may also be changed to account for this added precision to the offset. [0065] Another problem in explicit weighted prediction is that unidirectional prediction and bidirectional prediction may share the same weights and offsets. In order to have more flexibility for better prediction, unidirectional prediction and bidirectional prediction may be decoupled in accordance with this disclosure. In this case, unidirectional prediction and bidirectional prediction may define different weights and offsets for a given type of prediction (default, implicit or explicit). New syntax elements may be defined for explicit prediction to allow for better prediction. An encoder can include the syntax elements in a coded bitstream to signal different rounding modes used by the encoder, so that the same rounding modes can be used by the decoder. [0066] It is beneficial to adaptively select rounding adjustments. One way of doing this is to generate two or more different sets of predictive data (and possibly encode a video block several times) based on the two or more different sets of predictive data. One set of the predictive data may have nonzero rounding and the other may eliminate the rounding. In still other examples, upward rounding, downward rounding, and no rounding may be considered. Motion compensation unit 35 may generate these different types of predictive data, and may conduct a rate-distortion (RD) analysis to select the best predictive data for a given video block.

Nokia Exhibit 2006, p. 25
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0067] Rate-distortion (RD) analysis is fairly common in video coding, and generally involves the calculation of a cost metric indicative of the coding cost. The cost metric may balance the number of bits required for the coding (the rate) and the level of quality associated with the coding (the distortion). A typical rate-distortion cost calculation may generally correspond to the format of: $J(\lambda)=\lambda R+D$, where $J(\lambda)$ is the cost, R is the bit rate, D is the distortion, and λ is the Lagrange multiplier. [0068] One way for video encoder 50 to identify the most desired prediction data is to use motion estimation unit 32 to first find motion vectors, and then implement motion compensation unit 35 and adder 48 to calculate prediction errors with and without rounding adjustments. Motion compensation unit 35 may then select the prediction data that yields the least prediction error. The prediction error can be calculated by using the sum of absolute difference between the prediction data and the current data being encoded. [0069] In explicit weighted prediction, motion compensation unit 35 of video encoder 50 may implement three different modes. In all three explicit weighted prediction modes, each reference picture may have one offset used for unidirectional prediction and each pair of reference pictures has one offset for bidirectional prediction such as: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+(o<<r)+2^r)>>(r+1)$, where pred(i,j) is the first weighted prediction data, pred0(i,j) is data from list 0, pred1(i,j) is data from list 1, w0 and w1 are weight factors, o is a common offset applicable the pred0(i,j) from list 0 and the pred1(i,j) from list 1, and r and 2^r are rounding terms that provide the rounding in conjunction with a right shift operation ">>" by (r+1) bits. The first mode may use the weights defined by default weighted prediction. The second may use weights defined by

Nokia Exhibit 2006, p. 26
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	implicit weighted prediction. The third mode allows each reference picture to have one weight for unidirectional prediction and each pair of reference pictures involved in bidirectional prediction to have a pair of weights for both reference pictures. The weights defined for the third mode may be determined adaptively, and in some cases, the explicit weighted prediction framework may be used with weights defined by default or implicit weighted prediction in order to allow for offset in these contexts. Furthermore, the weights and offsets defined in this third mode may be different for unidirectional and bidirectional prediction. The equation above may also be modified slightly to provide higher precision for the offset. If higher precision for the offset is desired, the offset may be multiplied by a factor (such as by 2) and then rounded to an integer number. The left shift may also be changed to account for this added precision to the offset, e.g., in this case, the left shift may changed to r−1. [0070] In order for video encoder 50 to signal to a decoder the particular mode that was used for a given video block or set of video blocks, video encoder 50 may implement 2 single-bit syntax elements: derived_weight_flag and poc_weight_flag. In this case, the derived weight flag may be used to select between the first two explicit weighted prediction modes mentioned above, and the third mode and poc_weight_flag may be used to select between the first and the second explicit weighted prediction modes. [0071] In order to find the best weighted prediction, video encoder 50 may perform multi-pass encoding and select the best mode based on rate-distortion cost. One way to do this is an exhaustive search where motion compensation unit 35 generates every possible weighted prediction data and selects the best one. To reduce the complexity, however, motion compensation unit 35 may implement additional techniques of this disclosure, e.g., to first select between default weighted prediction and implicit weighted prediction, and then to compare the selection to explicit weighted prediction. Motion compensation unit 35 may calculate weights and offsets associated with explicit weighted prediction, and may also use the explicit weighted prediction framework to add offsets to data otherwise associated with default weighted prediction or implicit weighted prediction, whichever was selected. Thus,

Nokia Exhibit 2006, p. 27
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	there may be two sets of offsets calculated by motion compensation unit 35. The first set of offsets may be calculated by using known weights used in the default weighted prediction or the implicit weighted prediction, and the second set of offsets may be calculated jointly with weights as part of a normal calculation of explicit weighted prediction, e.g., by minimizing the motion compensated prediction error. [0072] To further reduce the complexity, during explicit weighted prediction, if the offsets are 0, motion compensation unit 35 may skip the explicit weighted prediction that uses default weights or implicit weights. Also, if the offsets are 0 and calculated weights have no change, motion compensation unit 35 may skip the typical explicit weighted prediction that uses calculated weights and offsets. [0073] Once the desired prediction data is identified by motion compensation unit 35, as described herein, 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. [0074] Quantization unit 40 quantizes the residual transform coefficients to further reduce bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 9-bit value may be rounded down to an 8-bit value during

Nokia Exhibit 2006, p. 28
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	quantization. In addition, quantization unit 40 may also quantize the different offsets for cases where offset is used. [0075] Following quantization, entropy coding unit 46 entropy codes the quantized transform coefficients. For example, entropy coding unit 46 may perform content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. Following the entropy coding by entropy coding unit 46, the encoded video may be transmitted to another device or archived for later transmission or retrieval. The coded bitstream may include entropy coded residual blocks, motion vectors for such blocks, and other syntax such as the syntax described herein. [0076] Inverse quantization unit 42 and inverse transform unit 44 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual block in the pixel domain, e.g., for later use as the reference block in the manner described above. Adder 51 adds the reconstructed residual block to the motion compensated prediction block produced by motion compensation unit 35 to produce a reconstructed video block for storage in memory 34. The reconstructed video block may be used by motion estimation unit 32 and motion compensation unit 35 as a reference block to inter-encode a block in a subsequent video frame. Karczewicz II, as evidenced by the example citations below, discloses a method for encoding a block of pixels. See, e.g., Karczewicz II, Abstract: 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

Nokia Exhibit 2006, p. 29
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0002: This disclosure relates to digital video coding and, more particularly, fractional interpolations of predictive data used in video coding. Karczewicz II, ¶0004: 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.

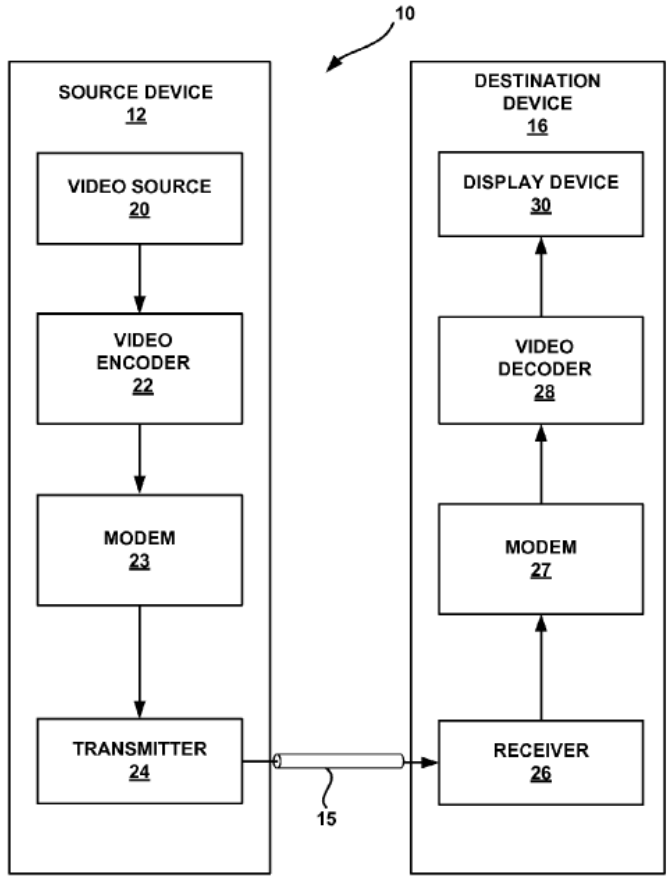
Nokia Exhibit 2006, p. 30
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0012: 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. Karczewicz II, ¶0014: 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. Karczewicz II, ¶0035: 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 2006, p. 31
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, Fig.1:

Nokia Exhibit 2006, p. 32
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 1

Nokia Exhibit 2006, p. 33
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0040, 41, 43: [0040] FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite radiotelephones, or any wireless devices that can communicate video information over a communication channel 15, in which case communication channel 15 is wireless. The techniques of this disclosure, however, which concern motion compensated interpolation, are not necessarily limited to wireless applications or settings. [0041] 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. [0043] Video source 20 of source device 12 may include a video capture device, such as a video camera, a video archive containing previously captured video, or a video feed from a video content provider. As a further alternative, video source 20 may generate computer graphics-based data as the source video, or a combination of live video, archived video, and computer-generated video. In some cases, if video source 20 is a video camera, source device 12 and destination device 16 may form so-called camera phones or video phones. In each case, the captured, pre-captured or computer-generated video may be encoded by video encoder 22. The encoded video information may then be modulated by modem 23 according to a

Nokia Exhibit 2006, p. 34
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	communication standard, e.g., such as code division multiple access (CDMA) or another communication standard, and transmitted to destination device 16 via transmitter 24. Modem 23 may include various mixers, filters, amplifiers or other components designed for signal modulation. For wireless applications, transmitter 24 may include circuits designed for transmitting data, including amplifiers, filters, and one or more antennas. Karczewicz II, Fig. 2:

Nokia Exhibit 2006, p. 35
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

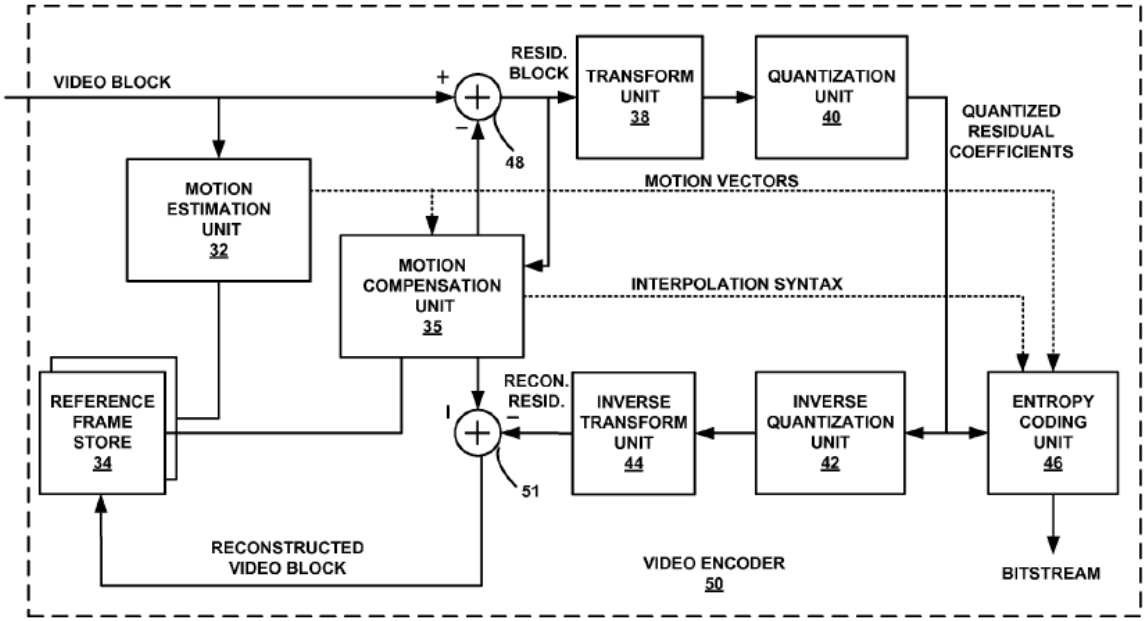


FIG. 2

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 H.264

Nokia Exhibit 2006, p. 36
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0050: 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. Karczewicz II, ¶¶0054-62: [0054] FIG. 2 is a block diagram illustrating an example of a video encoder 50 that may perform motion compensated interpolation consistent with this disclosure. Video encoder 50 may correspond to video encoder 22 of device 20, or a video encoder of a different device. Video encoder 50 may perform intra- and inter-coding of blocks within video frames, although intra-coding is not illustrated. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames of a video sequence. Intra-mode (I-mode) may refer to the spatial based compression mode and Inter-modes such as

Nokia Exhibit 2006, p. 37
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	prediction (P-mode) or bi-directional (B-mode) may refer to the temporal based compression modes. The techniques of this disclosure typically may be applied during inter-coding, and therefore, intra-coding units such as spatial prediction unit are not illustrated in FIG. 2 for simplicity and ease of illustration. However, the rounding techniques of this disclosure may also be applicable to spatial prediction and intra coding techniques. [0055] As shown in FIG. 2, video encoder 50 receives a current video block within a video frame to be encoded. In the example of FIG. 2, video encoder 50 includes motion estimation unit 32, a motion compensation unit 35, a reference frame store 34, an adder 48, a transform unit 38, a quantization unit 40, and an entropy coding unit 46. For video block reconstruction, video encoder 50 also includes an inverse quantization unit 42, an inverse transform unit 44 adder 51. A deblocking filter (not shown) may also be included to filter block boundaries to remove blockiness artifacts from reconstructed video. If desired, the deblocking filter would typically filter the output of adder 51. [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 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.

Nokia Exhibit 2006, p. 38
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0057] 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. [0058] 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 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. [0059] 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

Nokia Exhibit 2006, p. 39
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. [0060] 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. [0061] Quantization unit 40 quantizes the residual transform coefficients to further reduce bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 16-bit value may be rounded down to a 15-bit value during quantization. Following quantization, entropy coding unit 46 entropy codes the quantized transform coefficients. For example, entropy coding unit 46 may perform content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. Following the entropy coding by entropy coding unit 46, the encoded video may be transmitted to another device or archived for later transmission or retrieval. The coded bitstream may include entropy coded residual blocks, motion vectors for

Nokia Exhibit 2006, p. 40
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	such blocks, and other syntax including the interpolation syntax that identifies the interpolation filters that were applied by motion compensation unit 35. [0062] Inverse quantization unit 42 and inverse transform unit 44 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual block in the pixel domain, e.g., for later use a reference block. Summer 51 adds the reconstructed residual block to the motion compensated prediction block produced by motion compensation unit 35 to produce a reconstructed video block for storage in reference frame store 34. The reconstructed video block may be used by motion estimation unit 32 and motion compensation unit 35 as a reference block to inter-encode a block in a subsequent video frame. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth supra. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 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 I, 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. *See, e.g.*, Karczewicz I, ¶0004:

Nokia Exhibit 2006, p. 41
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	In video encoding, the compression often includes spatial prediction, motion estimation and motion compensation. Intra-coding relies on spatial prediction and transform coding, such as discrete cosine transform (DCT), to reduce or remove spatial redundancy between video blocks within a given video frame. Inter-coding relies on temporal prediction and transform coding to reduce or remove temporal redundancy between video blocks of successive video frames of a video sequence. Intra-coded frames ("I-frames") are often used as random access points as well as references for the inter-coding of other frames. I-frames, however, typically exhibit less compression than other frames. The term I-units may refer to I-frames, I-slices or other independently decodable portions of an I-frame. Karczewicz I, ¶0005: For inter-coding, a video encoder performs motion estimation to track the movement of matching video blocks between two or more adjacent frames or other coded units, such as slices of frames. Inter-coded frames may include predictive frames ("P-frames"), which may include blocks predicted from a previous frame, and bidirectional predictive frames ("B-frames"), which may include blocks predicted from a previous frame and a subsequent frame of a video sequence. The terms P-frames and B-frames are somewhat historic in the sense that early coding techniques limited prediction in specific directions. Newer coding formats and standards may not limit the prediction direction of P-frames or B-frames. Thus, the term "bi-directional" now refers to prediction based on two or more lists of reference data regardless of the temporal relationship of such reference data relative to the data being coded. Karczewicz I, ¶0006: Consistent with newer video standards such as ITU H.264, for example, bi-directional prediction may be based on two different lists which do not necessarily need to have data that resides temporally before and after the current video block. In other words, B-video blocks may be predicted from two lists of data, which may correspond to data from two previous frames,

Nokia Exhibit 2006, p. 42
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	two subsequent frames, or one previous frame and one subsequent frame. In contrast, P-video blocks are predicted based on one list, i.e., one data structure, which may correspond to one predictive frame, e.g., one previous frame or one subsequent frame. B-frames and P-frames may be more generally referred to as P-units and B-units. P-units and B-units may also be realized in smaller coded units, such as slices of frames or portions of frames. B-units may include B-video blocks, P-video blocks or I-video blocks. P-units may include P-video blocks or I-video blocks. I-units may include only I-video blocks. Karczewicz I, ¶0007: For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0022: This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block is predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, rounding adjustments to bi-directional predictive data may be purposely eliminated to provide predictive data that lacks any rounding bias. In this case, rounded and unrounded predictive data may both be considered in a rate-distortion analysis to identify the best data for prediction of a given video block. One or more syntax elements may be encoded to indicate the selection,

Nokia Exhibit 2006, p. 43
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	and a decoder may interpret the one or more syntax elements in order to determine whether rounding should be used in the decoding process. Karczewicz I, ¶0042: The techniques of this disclosure are specifically applicable to weighted bi-directional prediction. As mentioned above, bi-directional prediction is prediction of so-called "B-video blocks" based on two different lists of data. B-video blocks may be predicted from two lists of data from two previous frames, two lists of data from subsequent frames, or one list of data from a previous frame and one from a subsequent frame. In contrast, P-video blocks are predicted based on one list, which may correspond to one predictive frame, e.g., one previous frame or one subsequent frame. B-frames and P-frames may be more generally referred to as P-units and B-units. P-units and B-units may also be realized in smaller coded units, such as slices of frames or portions of frames. B-units may include B-video blocks, P-video blocks or I-video blocks. P-units may include P-video blocks or I-video blocks. I-units may include only I-video blocks. Karczewicz I, Fig. 2:

Nokia Exhibit 2006, p. 44
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

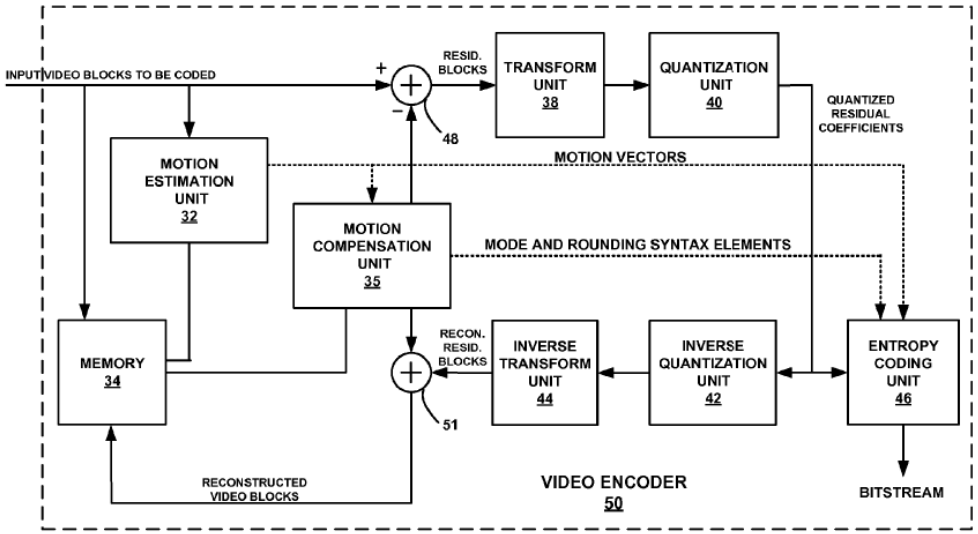


FIG. 2

Karczewicz I, ¶0051:

> FIG. 2 is a block diagram illustrating an example of a video encoder 50 that may perform techniques consistent with this disclosure. Video encoder 50 may correspond to video encoder 22 of source device 12, or a video encoder of a different device. Video encoder 50 may perform intra- and inter-coding of blocks within video frames, although intra-coding components are not shown in FIG. 2 for ease of illustration. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames of a video sequence. Intra-mode (I-mode) may refer to the spatial based compression mode and

Nokia Exhibit 2006, p. 45
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Inter-modes such as prediction (P-mode) or bi-directional (B-mode) may refer to the temporal based compression modes. Karczewicz I, ¶0053: 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 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. For demonstrative purposes, the techniques described in this disclosure are described as being performed by motion compensation unit 35. Karczewicz I, ¶0054: Motion estimation unit 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of one or more predictive coded units (e.g., a previous and/or future frame in terms of time or temporally). Motion estimation unit 32 may, as an example, select a motion vector for a B-frame in a number of ways. In one way, motion estimation unit 32 may select a previous or future frame from a first set of frames (referred to as list 0) and determine a motion vector using only this previous or future frame from list 0. Alternatively, motion estimation unit 32 may select a previous or future frame from a second set of frames (referred to as list 1) and determine a motion vector using only this previous or future frame from list 1. In yet another way, motion estimation unit 32 may select a first frame

Nokia Exhibit 2006, p. 46
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	from list 0 and a second frame from list 1 and select one or more motion vectors from the first frame of list 0 and the second frame of list 1. This third form of prediction may be referred to as bi-predictive motion estimation. Techniques of this disclosure may be implemented so as to efficiently select a motion-compensated bi-prediction mode. The selected motion vector for any given list may point to a predictive video block that is most similar to the video block being coded, e.g., as defined by a metric such as sum of absolute difference (SAD) or sum of squared difference (SSD) of pixel values of the predictive block relative to pixel values of the block being coded. 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. Karczewicz II, ¶0004: 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 2006, p. 47
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0006: 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. Karczewicz II, Fig. 2:

Nokia Exhibit 2006, p. 48
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

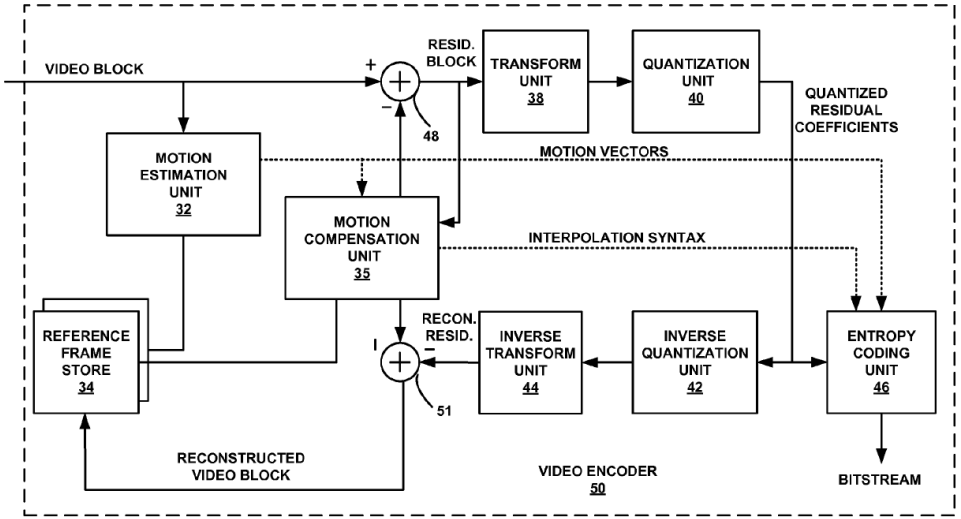


FIG. 2

Karczewicz II, ¶0054:

> FIG. 2 is a block diagram illustrating an example of a video encoder 50 that may perform motion compensated interpolation consistent with this disclosure. Video encoder 50 may correspond to video encoder 22 of device 20, or a video encoder of a different device. Video encoder 50 may perform intra- and inter-coding of blocks within video frames, although intra-coding is not illustrated. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames of a video sequence. Intra-mode (I-mode) may refer to the spatial based compression mode and Inter-modes such as prediction (P-mode) or bi-directional (B-mode) may refer to the temporal based compression

Nokia Exhibit 2006, p. 49
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	modes. The techniques of this disclosure typically may be applied during inter-coding, and therefore, intra-coding units such as spatial prediction unit are not illustrated in FIG. 2 for simplicity and ease of illustration. However, the rounding techniques of this disclosure may also be applicable to spatial prediction and intra coding techniques. 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 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. Karczewicz II, ¶0057: 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,

Nokia Exhibit 2006, p. 50
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0058: 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 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. Karczewicz II, ¶0073: 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.

Nokia Exhibit 2006, p. 51
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0089: 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). 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 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 2006, p. 52
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 53
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 I, 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. Karczewicz I, ¶0007: > For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0024: > Weighted prediction refers to bi-directional prediction in which weights can be assigned to two or more different sets of predictive data. Default weighted prediction refers to weighted prediction in which the weight factors associated with two or more different lists are pre-

Nokia Exhibit 2006, p. 54
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	defined by some default setting. Default weighted prediction, in some cases, may assign equal weightings to each of the lists. Implicit weighted prediction refers to weighted prediction in which the weight factors associated with two or more different lists are defined based on some implicit factors associated with the data. For example, implicit weight factors may be defined by the temporal positions of the data in the two different lists relative to the data being predictively coded. If data in list 0 is temporally closer to the data being predictively coded than data in list 1, then the data is list 0 may be assigned a greater implicit weigh factor in implicit weighted prediction. Karczewicz I, ¶0041: Following inter-based predictive encoding (which includes interpolation and the techniques of this disclosure to efficiently select a prediction algorithm or mode by which to predict a coded unit), and following any transforms (such as the 4×4 or 8×8 integer transform used in H.264/AVC or a discrete cosine transform or DCT), quantization may be performed. Quantization generally refers to a process in which coefficients are quantized to possibly reduce the amount of data used to represent the coefficients. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 16-bit value may be rounded down to a 15-bit value during quantization. Following quantization, entropy coding may be performed, e.g., according to content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. Karczewicz I, ¶0044: For different types of weighted bi-directional prediction in accordance with ITU-T H.264, video encoder 22 and video decoder 28 may generally support three different types of prediction modes. A first prediction mode, referred to as "default weighted prediction," refers to weighted prediction in which the weight factors associated with two or more different lists

Nokia Exhibit 2006, p. 55
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	are pre-defined by some default setting. Default weighted prediction, in some cases, may assign equal weightings to each of the lists. Karczewicz I, ¶0048: In accordance with another aspect of this disclosure, video encoder 22 may select among default weighted prediction, implicit weighted prediction, and explicit weighted prediction. In this aspect, rather than consider each of these possibilities relative to one another, encoder 22 may be programmed to select between default weighted prediction and implicit weighted prediction. The selection, then, may be compared to explicit weighted prediction. Specifically, encoder 22 may perform explicit weighted prediction to calculate explicit weight factors, but encoder 22 may also compare the computed explicit weighted prediction to explicit weighted prediction having weight factors that correspond to those defined by either default weighted prediction or implicit weighted prediction. Karczewicz I, ¶0055: According to the ITU-T H.264/AVC standard, three motion-compensated bi-predictive algorithms or modes may be used to predict a B-frame or portions thereof, such as video blocks, macroblocks or any other discreet and/or contiguous portion of a B-frame. A first motion-compensated bi-predictive algorithm or mode, which is commonly referred to as default weighted prediction, may involve applying default weights to each identified video block of the first frame of list 0 and the second frame of list 1. The default weights may be programmed according to the standard, and are often selected to be equal for default weighted prediction. The weighted blocks of the first and second frames are then added together and divided by the total number of frames used to predict the B-frame, e.g., two in this instance. Often, this division is accomplished by adding 1 to the addition of the weighted blocks of the first and

Nokia Exhibit 2006, p. 56
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	second frames and then shifting the result to the right by one bit. The addition of 1 is a rounding adjustment. Karczewicz I, ¶0053: 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 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. For demonstrative purposes, the techniques described in this disclosure are described as being performed by motion compensation unit 35. Karczewicz I, ¶0054: Motion estimation unit 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of one or more predictive coded units (e.g., a previous and/or future frame in terms of time or temporally). Motion estimation unit 32 may, as an example, select a motion vector for a B-frame in a number of ways. In one way, motion estimation unit 32 may select a previous or future frame from a first set of frames (referred to as list 0) and determine a motion vector using only this previous or future frame from list 0. Alternatively, motion estimation unit 32 may select a previous or future frame from a second set of frames (referred to as list 1) and determine a motion vector using only this previous or future frame from list 1. In yet another way, motion estimation unit 32 may select a first frame

Nokia Exhibit 2006, p. 57
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	from list 0 and a second frame from list 1 and select one or more motion vectors from the first frame of list 0 and the second frame of list 1. This third form of prediction may be referred to as bi-predictive motion estimation. Techniques of this disclosure may be implemented so as to efficiently select a motion-compensated bi-prediction mode. The selected motion vector for any given list may point to a predictive video block that is most similar to the video block being coded, e.g., as defined by a metric such as sum of absolute difference (SAD) or sum of squared difference (SSD) of pixel values of the predictive block relative to pixel values of the block being coded. Karczewicz I, ¶0057: More generally, weighted prediction may be given by: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+2^r)>>(r+1)$ where pred(i,j) is data associated with the weighed prediction block, pred0(i,j) is data from list 0, pred1(i,j) is data from list 1, w0 and w1 are the weight factors, 2^r is the rounding term, and >> is a right shift operation by (r+1) bits. Consistent with this disclosure, two different version of pred(i,j) may be generated and considered by motion compensation unit 35. The first is consistent with the equation above, and the second is consistent with the equation above without rounding, i.e., with the term "2r" removed from the equation. Eliminating this rounding may achieve better weighted predictive data in some cases, which can improve coding efficiency. Motion compensation unit 35 may generate one or more syntax elements to define whether or not rounding was used for a given video block or set of video blocks. Both the bi-directional prediction mode and the one or more syntax elements indicating whether rounding was used may be output form motion compensation unit 35 to entropy coding unit 46 for inclusion in the encoded bitstream.

Nokia Exhibit 2006, p. 58
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, ¶0058: B pictures use two lists of previously-coded reference pictures, list 0 and list 1. These two lists can each contain past and/or future coded pictures in temporal order. Blocks in a B picture may be predicted in one of several ways: motion-compensated prediction from a list 0 reference picture, motion-compensated prediction from a list 1 reference picture, or motion-compensated prediction from the combination of both list 0 and list 1 reference pictures. To get the combination of both list 0 and list 1 reference pictures, two motion compensated reference areas are obtained from list 0 and list 1 reference picture respectively. Their combination will be used to predict the current block. Karczewicz I, ¶0060: Default weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: pred(i,j)=pred0(i,j) Bidirectional prediction: pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1 where pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1. 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. *See, e.g.,*

Nokia Exhibit 2006, p. 59
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0010: 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. Karczewicz II, ¶¶0038-39: [0038] 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.

Nokia Exhibit 2006, p. 60
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0039] This disclosure recognizes coding inefficiencies that can occur due to conventional rounding of half-pixel values, and proposes techniques to improve interpolation by eliminating or reducing 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 may be applied to some of the half-pixel values in order to ensure that a sixteen-bit data structure can be used to store any intermediate values. In some cases, the impact of the slight rounding on the final accuracy can be zero by performing the rounding properly. In any case, the elimination or reduction of intermediate rounding may be done with or without the implementation of a plurality of pre-defined interpolation filters, as described herein. Karczewicz II, ¶0053: 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 2006, p. 61
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0059: 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. Karczewicz II, ¶¶0093-0094: 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: b1=C1−5*C2+20*C3+20*C4−5*C5+C6 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. H1=A3−5*B3+20*C3+20*D3−5*E3+F3

Nokia Exhibit 2006, p. 62
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. In this case, the values of "b" and "h" may be defined as: b=max(0, min(255, (b1+16)>>5)) h=max(0, min(255, (h1+16)>>5)) where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation. 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: j=max(0, min(255, (j1+512)>>10)).

Nokia Exhibit 2006, p. 63
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0096: 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. 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, ¶¶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.

Nokia Exhibit 2006, p. 64
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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, ¶¶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 2006, p. 65
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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, ¶¶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.

Nokia Exhibit 2006, p. 66
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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.

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 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 2006, p. 67
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 68
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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, ¶0104-0106:

> 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 2006, p. 69
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 71
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 72
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, Figs. 4A-4D:

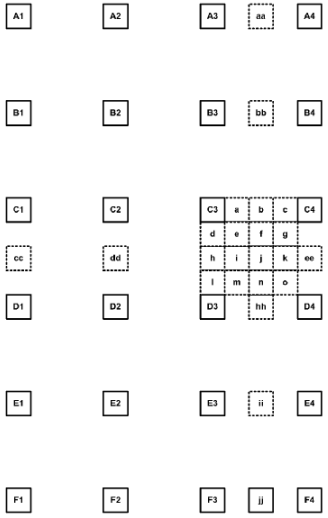


FIG. 4A

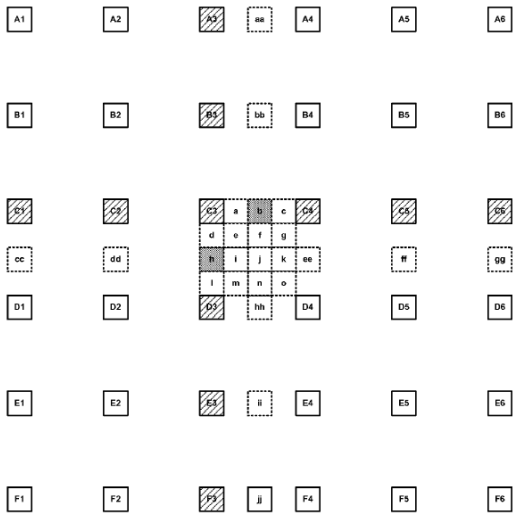


FIG. 4B

Nokia Exhibit 2006, p. 73
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

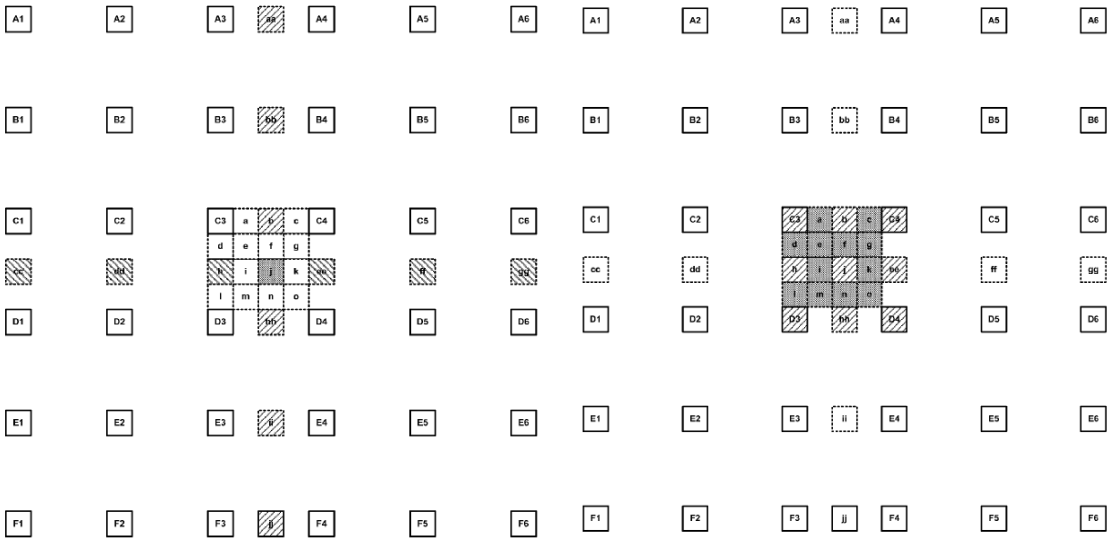


FIG. 4C

FIG. 4D

See also the disclosures of limitation 1[a], reproduced *supra*.

See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*.

Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 74
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
1[c] using said second reference block to obtain a second prediction, said second prediction having the second precision;	Karczewicz I, 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. Karczewicz I, ¶0007: > For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0024: > Weighted prediction refers to bi-directional prediction in which weights can be assigned to two or more different sets of predictive data. Default weighted prediction refers to weighted prediction in which the weight factors associated with two or more different lists are pre-defined by some default setting. Default weighted prediction, in some cases, may assign equal weightings to each of the lists. Implicit weighted prediction refers to weighted prediction in which the weight factors associated with two or more different lists are defined based on some implicit factors associated with the data. For example, implicit weight factors may be defined by the temporal positions of the data in the two different lists relative to the data being predictively coded. If data in list 0 is temporally closer to the data being predictively coded than data in list 1, then the data is list 0 may be assigned a greater implicit weigh factor in implicit weighted prediction.

Nokia Exhibit 2006, p. 75
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, ¶0041: Following inter-based predictive encoding (which includes interpolation and the techniques of this disclosure to efficiently select a prediction algorithm or mode by which to predict a coded unit), and following any transforms (such as the 4×4 or 8×8 integer transform used in H.264/AVC or a discrete cosine transform or DCT), quantization may be performed. Quantization generally refers to a process in which coefficients are quantized to possibly reduce the amount of data used to represent the coefficients. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 16-bit value may be rounded down to a 15-bit value during quantization. Following quantization, entropy coding may be performed, e.g., according to content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. Karczewicz I, ¶0044: For different types of weighted bi-directional prediction in accordance with ITU-T H.264, video encoder 22 and video decoder 28 may generally support three different types of prediction modes. A first prediction mode, referred to as "default weighted prediction," refers to weighted prediction in which the weight factors associated with two or more different lists are pre-defined by some default setting. Default weighted prediction, in some cases, may assign equal weightings to each of the lists. Karczewicz I, ¶0048: In accordance with another aspect of this disclosure, video encoder 22 may select among default weighted prediction, implicit weighted prediction, and explicit weighted prediction. In this aspect, rather than consider each of these possibilities relative to one another, encoder 22 may be programmed to select between default weighted prediction and implicit weighted prediction. The selection, then, may be compared to explicit weighted prediction. Specifically,

Nokia Exhibit 2006, p. 76
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	encoder 22 may perform explicit weighted prediction to calculate explicit weight factors, but encoder 22 may also compare the computed explicit weighted prediction to explicit weighted prediction having weight factors that correspond to those defined by either default weighted prediction or implicit weighted prediction. Karczewicz I, ¶0055: According to the ITU-T H.264/AVC standard, three motion-compensated bi-predictive algorithms or modes may be used to predict a B-frame or portions thereof, such as video blocks, macroblocks or any other discreet and/or contiguous portion of a B-frame. A first motion-compensated bi-predictive algorithm or mode, which is commonly referred to as default weighted prediction, may involve applying default weights to each identified video block of the first frame of list 0 and the second frame of list 1. The default weights may be programmed according to the standard, and are often selected to be equal for default weighted prediction. The weighted blocks of the first and second frames are then added together and divided by the total number of frames used to predict the B-frame, e.g., two in this instance. Often, this division is accomplished by adding 1 to the addition of the weighted blocks of the first and second frames and then shifting the result to the right by one bit. The addition of 1 is a rounding adjustment. Karczewicz I, ¶0057: More generally, weighted prediction may be given by: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+2^r)>>(r+1)$ where pred(i,j) is data associated with the weighed prediction block, pred0(i,j) is data from list 0, pred1(i,j) is data from list 1, w0 and w1 are the weight factors, 2^r is the rounding term, and >> is a right shift operation by (r+1) bits. Consistent with this disclosure, two different version

Nokia Exhibit 2006, p. 77
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	of pred(i,j) may be generated and considered by motion compensation unit 35. The first is consistent with the equation above, and the second is consistent with the equation above without rounding, i.e., with the term "2r" removed from the equation. Eliminating this rounding may achieve better weighted predictive data in some cases, which can improve coding efficiency. Motion compensation unit 35 may generate one or more syntax elements to define whether or not rounding was used for a given video block or set of video blocks. Both the bi-directional prediction mode and the one or more syntax elements indicating whether rounding was used may be output form motion compensation unit 35 to entropy coding unit 46 for inclusion in the encoded bitstream. Karczewicz I, ¶0058: B pictures use two lists of previously-coded reference pictures, list 0 and list 1. These two lists can each contain past and/or future coded pictures in temporal order. Blocks in a B picture may be predicted in one of several ways: motion-compensated prediction from a list 0 reference picture, motion-compensated prediction from a list 1 reference picture, or motion-compensated prediction from the combination of both list 0 and list 1 reference pictures. To get the combination of both list 0 and list 1 reference pictures, two motion compensated reference areas are obtained from list 0 and list 1 reference picture respectively. Their combination will be used to predict the current block. Karczewicz I, ¶0060: Default weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: pred(i,j)=pred0(i,j) Bidirectional prediction: pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1

Nokia Exhibit 2006, p. 78
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	where pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1. 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. *See, e.g.*, Karczewicz II, ¶0010: > 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. Karczewicz II, ¶0038: > 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

Nokia Exhibit 2006, p. 79
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0053: 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. Karczewicz II, ¶0059: 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

Nokia Exhibit 2006, p. 80
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶¶0093-0094: 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: b1=C1−5*C2+20*C3+20*C4−5*C5+C6 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. h1=A3−5*B3+20*C3+20*D3−5*E3+F3 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. In this case, the values of "b" and "h" may be defined as: b=max(0, min(255, (b1+16)>>5)) h=max(0, min(255, (h1+16)>>5))

Nokia Exhibit 2006, p. 81
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation. 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: j=max(0, min(255, (j1+512)>>10)). Karczewicz II, ¶0096: 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 2006, p. 82
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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, ¶¶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.

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).

Nokia Exhibit 2006, p. 83
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. 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 2006, p. 84
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 unrounded values denoted as "b1," "h1," "ee1," and "hh1" for purposes of quarter-pixel generation. 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

Nokia Exhibit 2006, p. 85
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 86
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 87
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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, ¶0104-106:

> 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."

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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	416	10s
r1 = max(0, r1)	clip r1 on the low side	0	416	9u
r1 = min(255, r1)	clip r1 on the high side	0	255	8u

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 90
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

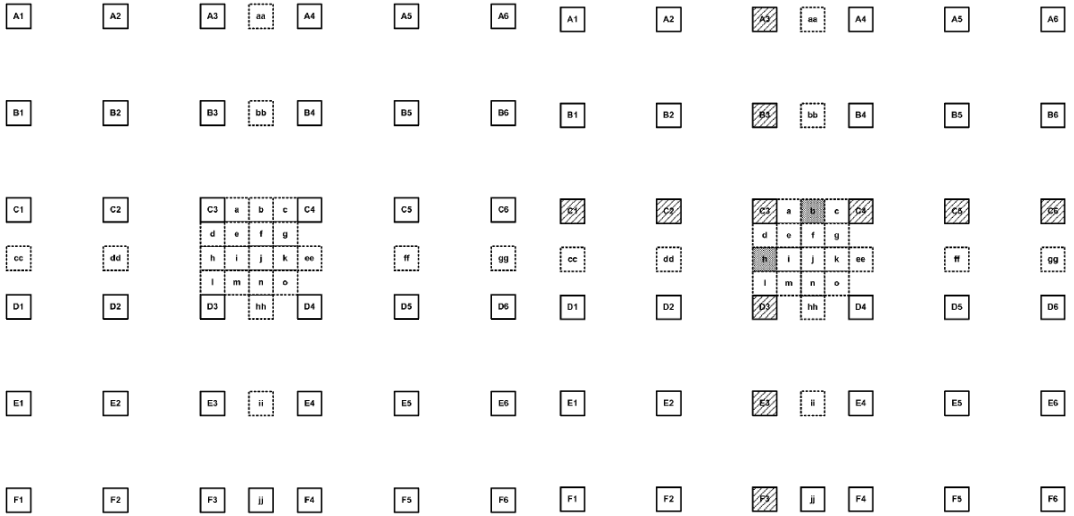
TABLE 8-continued

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 2006, p. 91
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, Figs. 4A-4D:  FIG. 4A FIG. 4B

Nokia Exhibit 2006, p. 92
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

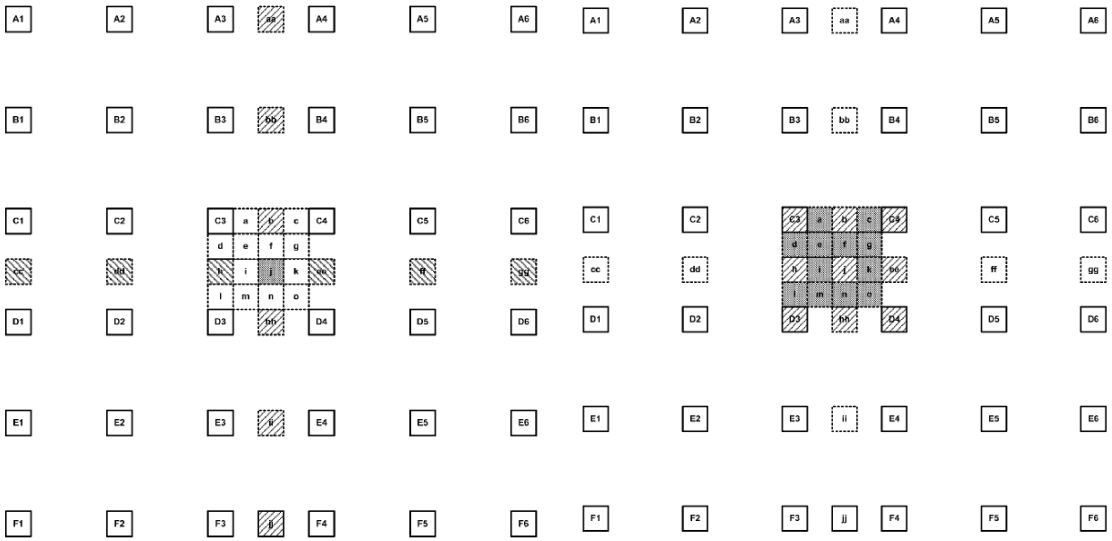


FIG. 4C

FIG. 4D

See also the disclosures of limitation 1[b], reproduced *supra*.

See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*.

Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 93
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
1[d] obtaining a combined prediction based at least partly upon said first prediction and said second prediction;	Karczewicz I, as evidenced by the example citations below, discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction. *See, e.g.,* Karczewicz I, ¶0060: > Default weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. > Unidirectional prediction: pred(i,j)=pred0(i,j) > Bidirectional prediction: pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1 > where pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1. 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. *See, e.g.,* Karczewicz II, ¶0054: > FIG. 2 is a block diagram illustrating an example of a video encoder 50 that may perform motion compensated interpolation consistent with this disclosure. Video encoder 50 may correspond to video encoder 22 of device 20, or a video encoder of a different device. Video encoder 50 may perform intra- and inter-coding of blocks within video frames, although intra-coding is not illustrated. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames of a video sequence.

Nokia Exhibit 2006, p. 94
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Intra-mode (I-mode) may refer to the spatial based compression mode and Inter-modes such as prediction (P-mode) or bi-directional (B-mode) may refer to the temporal based compression modes. The techniques of this disclosure typically may be applied during inter-coding, and therefore, intra-coding units such as spatial prediction unit are not illustrated in FIG. 2 for simplicity and ease of illustration. However, the rounding techniques of this disclosure may also be applicable to spatial prediction and intra coding techniques. Karczewicz II, ¶¶0093-0106: 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: b1=C1−5*C2+20*C3+20*C4−5*C5+C6 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. h1=A3−5*B3+20*C3+20*D3−5*E3+F3 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. In this case, the values of "b" and "h" may be defined as: b=max(0, min(255, (b1+16)>>5))

Nokia Exhibit 2006, p. 95
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	h=max(0, min(255, (h1+16)>>5)) where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation. 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)). 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. 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.

Nokia Exhibit 2006, p. 96
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 97
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. 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

Nokia Exhibit 2006, p. 98
ASUS v. Nokia
IPR2025-01154

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

TABLE 5

positions {s, e, 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 2006, p. 99
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 100
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 101
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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, eel 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

Nokia Exhibit 2006, p. 102
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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." See also the disclosures of limitations 1[b] and 1[c], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 I, 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. Karczewicz I, ¶0060: Default weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: pred(i,j)=pred0(i,j) Bidirectional prediction: pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1 where pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1.

Nokia Exhibit 2006, p. 103
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, ¶0055: According to the ITU-T H.264/AVC standard, three motion-compensated bi-predictive algorithms or modes may be used to predict a B-frame or portions thereof, such as video blocks, macroblocks or any other discreet and/or contiguous portion of a B-frame. A first motion-compensated bi-predictive algorithm or mode, which is commonly referred to as default weighted prediction, may involve applying default weights to each identified video block of the first frame of list 0 and the second frame of list 1. The default weights may be programmed according to the standard, and are often selected to be equal for default weighted prediction. The weighted blocks of the first and second frames are then added together and divided by the total number of frames used to predict the B-frame, e.g., two in this instance. Often, this division is accomplished by adding 1 to the addition of the weighted blocks of the first and second frames and then shifting the result to the right by one bit. The addition of 1 is a rounding adjustment. Karczewicz I, ¶0057: More generally, weighted prediction may be given by: $pred(i,j)=(pred0(i,j)*w0+pred1(i,j)*w1+2^r)>>(r+1)$ where pred(i,j) is data associated with the weighed prediction block, pred0(i,j) is data from list 0, pred1(i,j) is data from list 1, w0 and w1 are the weight factors, 2^r is the rounding term, and >> is a right shift operation by (r+1) bits. Consistent with this disclosure, two different version of pred(i,j) may be generated and considered by motion compensation unit 35. The first is consistent with the equation above, and the second is consistent with the equation above without rounding, i.e., with the term "2r" removed from the equation. Eliminating this rounding may achieve better weighted predictive data in some cases, which can improve coding efficiency. Motion compensation unit 35 may generate one or more syntax elements to define

Nokia Exhibit 2006, p. 104
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	whether or not rounding was used for a given video block or set of video blocks. Both the bi-directional prediction mode and the one or more syntax elements indicating whether rounding was used may be output form motion compensation unit 35 to entropy coding unit 46 for inclusion in the encoded bitstream. 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. *See, e.g.*, Karczewicz II, ¶0094: In this case, the values of "b" and "h" may be defined as: b=max(0, min(255, (b1+16)>>5)) h=max(0, min(255, (h1+16)>>5)) where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation. *See also, e.g.*, ¶¶0094-0103 (using the ">>" right shift operation). In this case, the values of "b" and "h" may be defined as: b=max(0, min(255, (b1+16)>>5)) h=max(0, min(255, (h1+16)>>5)) where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation.

Nokia Exhibit 2006, p. 105
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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)). 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. 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

Nokia Exhibit 2006, p. 106
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. 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 2006, p. 107
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 108
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 109
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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:

> 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 2006, p. 110
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 111
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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

See also the disclosures of limitations 1[b], 1[c], and 1[d], reproduced *supra*.

See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*.

Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 112
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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 I, 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. Karczewicz I, ¶0007: > For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0073: > Once the desired prediction data is identified by motion compensation unit 35, as described herein, 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

Nokia Exhibit 2006, p. 113
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz I, ¶0074: Quantization unit 40 quantizes the residual transform coefficients to further reduce bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 9-bit value may be rounded down to an 8-bit value during quantization. In addition, quantization unit 40 may also quantize the different offsets for cases where offset is used. Karczewicz I, ¶0075: Following quantization, entropy coding unit 46 entropy codes the quantized transform coefficients. For example, entropy coding unit 46 may perform content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. Following the entropy coding by entropy coding unit 46, the encoded video may be transmitted to another device or archived for later transmission or retrieval. The coded bitstream may include entropy coded residual blocks, motion vectors for such blocks, and other syntax such as the syntax described herein. Karczewicz I, ¶0084: Video decoder 70 may receive encoded video data, and one or more syntax elements that indicate whether a rounding adjustment was used to encode the encoded video data. MC unit 86 of prediction unit 75 may generate weighted prediction data that depends on two or more lists of data, as described herein. In accordance with this disclosure, the weighted prediction data does not include the rounding adjustment if the one or more syntax elements indicate that the rounding adjustment was not used to encode the encoded video data. Video decoder 70 can

Nokia Exhibit 2006, p. 114
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	decode the video data using the weighted prediction data, e.g., by invoking adder 79 to add the weighted prediction data (e.g., a prediction block) to residual data (e.g., a residual block). Karczewicz I, ¶0089: Prediction unit 75 invokes motion compensation unit 86 for block based predictive decoding. In doing so, motion compensation unit 86 generates weighted prediction data based on the syntax (113). Accordingly, if the one or more syntax elements indicate that a rounding adjustment was used, then motion compensation unit 86 generates weighted prediction data that includes the rounding adjustment. However, if the one or more syntax elements indicate that a rounding adjustment was not used, then motion compensation unit 86 generates weighted prediction data that lacks the rounding adjustment. Video decoder 70 can then decode the video data using the weighted prediction data (114). In particular, video decoder 70 may invoke adder 79 to combine weighted prediction data (e.g., a prediction block) with residual video data (e.g., a residual block) in order to generate a reconstruction of the video data (e.g., a reconstructed video block). Karczewicz I, Figure 2 and related disclosures, reproduced *supra*. 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. *See, e.g.,* Karczewicz II, ¶0004: 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

Nokia Exhibit 2006, p. 115
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0006: 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. Karczewicz II, ¶0035: 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

Nokia Exhibit 2006, p. 116
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0050: 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. Karczewicz II, ¶0058: 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

Nokia Exhibit 2006, p. 117
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0060: 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. Karczewicz II, ¶0073: 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

Nokia Exhibit 2006, p. 118
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0005: 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). Karczewicz II, ¶0061: 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.

Nokia Exhibit 2006, p. 119
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	The transform may convert the residual information from a pixel domain to a frequency domain. Karczewicz II, ¶0110: 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. Karczewicz II, Fig. 2 and related disclosures, reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 120
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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 I, 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. Karczewicz I, ¶0041: > Following inter-based predictive encoding (which includes interpolation and the techniques of this disclosure to efficiently select a prediction algorithm or mode by which to predict a coded unit), and following any transforms (such as the 4×4 or 8×8 integer transform used in H.264/AVC or a discrete cosine transform or DCT), quantization may be performed. Quantization generally refers to a process in which coefficients are quantized to possibly reduce the amount of data used to represent the coefficients. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 16-bit value may be rounded down to a 15-bit value during quantization. Following quantization, entropy coding may be performed, e.g., according to content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. 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. *See, e.g.,* Karczewicz II, ¶0007: > 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

Nokia Exhibit 2006, p. 121
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0010: 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. Karczewicz II, ¶0042: 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

Nokia Exhibit 2006, p. 122
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0053: 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. Karczewicz II, ¶0059: 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

Nokia Exhibit 2006, p. 123
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0066: 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 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. Karczewicz II, ¶0067: 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." Karczewicz II, ¶0069:

Nokia Exhibit 2006, p. 124
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0070: 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 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. Karczewicz II, ¶¶0093-0106: 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: b1=C1−5*C2+20*C3+20*C4−5*C5+C6 where "C1," "C2," "C3," "C4," "C5" and "C6" represent the six closest integer pixels that

Nokia Exhibit 2006, p. 125
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. h1=A3−5*B3+20*C3+20*D3−5*E3+F3 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. In this case, the values of "b" and "h" may be defined as: b=max(0, min(255, (b1+16)>>5)) h=max(0, min(255, (h1+16)>>5)) where ">>" is a right-shift operation. In this disclosure, ">>" represents a right shift operation and "<<" represents a left shift operation. 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)). 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 2006, p. 126
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 127
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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.

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

Nokia Exhibit 2006, p. 128
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 129
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 130
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 131
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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

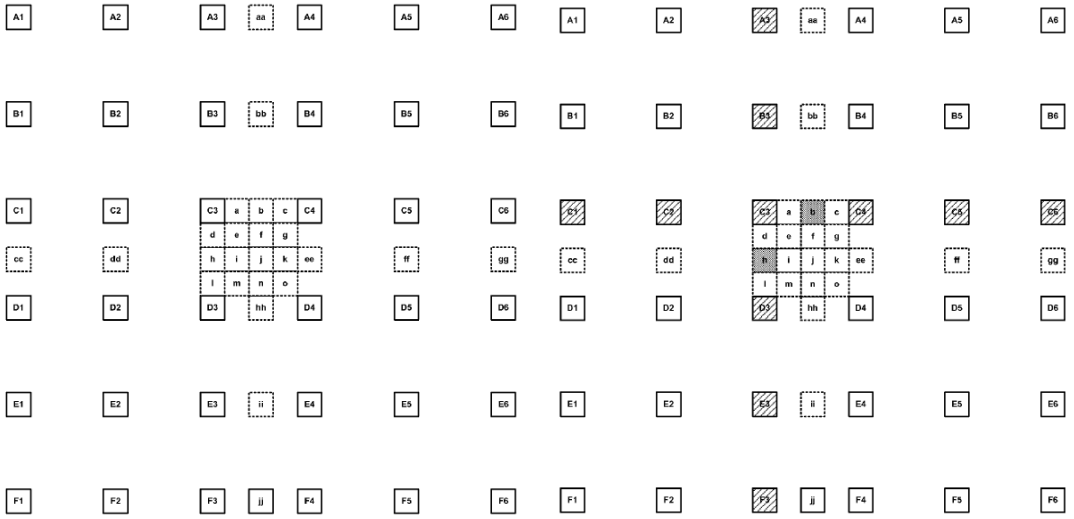
Operation	Comment	Min value	Max value	Register size
r2 = r2 >> 1	r2 is j1 >> 1	−4957	13116	15s
r1 = r1 + r2	r1 is y0 + (j1 >> 1)	−7507	23826	16s
r1 = r1 + 32	r1 is y0 + (j1 >> 1) + 32	−7491	23842	16s
r1 = r1 >> 6	r1 is (y0 + (j1 >> 1) + 32) >> 6	−235	745	11s
r1 = max (0, r1)	clip r1 on the low side	0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

Nokia Exhibit 2006, p. 133
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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." Karczewicz II, ¶0067: 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 2006, p. 134
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 4A FIG. 4B

Nokia Exhibit 2006, p. 135
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

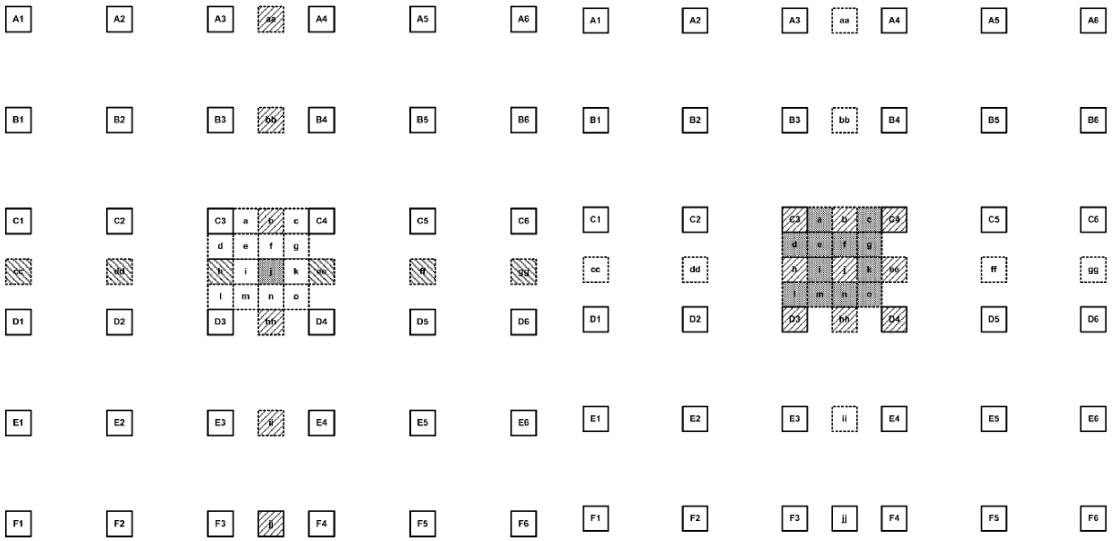


FIG. 4C

FIG. 4D

Karczewicz II, ¶¶0068-74:

> [0068] Every pixel location may have an associated fifteen different fractional locations, e.g., in compliance with interpolations compliant with the ITU H.264/AVC standard. In the example of FIGS. 4A-4D, these 15 different fractional locations associated with pixel "C3" are illustrated. For simplicity and ease of illustration, most of the other fractional locations are not shown (other than those mentioned above, which are used to generate one or more of the 15 different fractional locations associated with pixel "C3").

Nokia Exhibit 2006, p. 136
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0069] 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. [0070] 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 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. [0071] FIG. 4C illustrates one special case that may require two levels of interpolation. In particular, pixel location "j" is unlike the other half-pixel locations insofar as pixel location "j" is itself interpolated based on other half-pixel values. For example, pixel location "j" may be interpolated based on half-pixel interpolated values "cc," "dd," "h," ee," "ff," and "gg" in the horizontal direction. Alternatively, pixel location "j" may be interpolated based on half-pixel interpolated values "aa," "bb," "b," hh," "ii," and "jj" in the vertical direction. In FIG. 4C, interpolated pixel location "j" is shown with shading, half-pixel interpolated values "aa," "bb," "b," hh," "ii," and "jj" in the vertical direction are shown with right-to-left cross-hashes, and half-pixel interpolated values "cc," "dd," "h," ee," "ff," and "gg" are shown with left-to-right cross-hashes. [0072] FIG. 4D illustrates quarter-pixel locations "a," "c," "d," "e," f," "g," "i," "k," "l," m," "n," and "o" with shading, and illustrates the integer and half-pixel locations used for such

Nokia Exhibit 2006, p. 137
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	quarter-pixel interpolation (e.g., "C3," "b," "C4," "h," j," "ee," "D3," "hh," and "D4") with cross-hashes. [0073] 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. [0074] Again, FIGS. 4A-4D show the integer-pixel samples (also called full-pixels) in the solid boxes with upper-case letters. For any given integer-pixel sample, there are altogether 15 sub-pixel positions, which are shown for integer-pixel sample "C3" and labeled "a" through "o" in FIGS. 4A-4D. In H.264/AVC, the half-pixel positions "b," "h," and "j" may be calculated first using the one-dimensional 6-tap Wiener filter. As noted, half-pixel position "j" may require two levels of this interpolation using 6-tap filtering. In this case, for pixel position "j," the interpolation filter is applied first in the horizontal direction and then in the vertical direction, possibly with intermediate rounding of interpolated pixel values after interpolation in one dimension in order to ensure that data can be stored within sixteen-bit storage elements. Remaining quarter-pixel positions are then interpolated via filtering that uses a bilinear filter and the already calculated half-pixel samples. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

Nokia Exhibit 2006, p. 138
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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 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 I, 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. 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. *See, e.g.,* 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: j=max(0, min(255, (j1+512)>>10)). Karczewicz II, ¶0067: 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

Nokia Exhibit 2006, p. 139
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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."

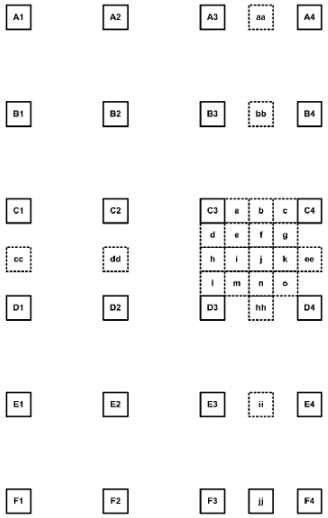


FIG. 4A

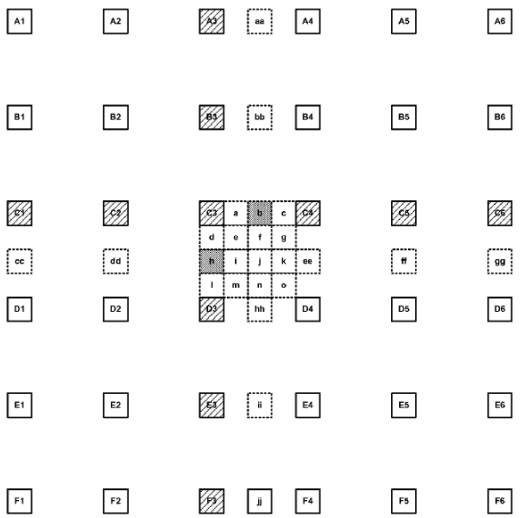


FIG. 4B

Nokia Exhibit 2006, p. 140
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

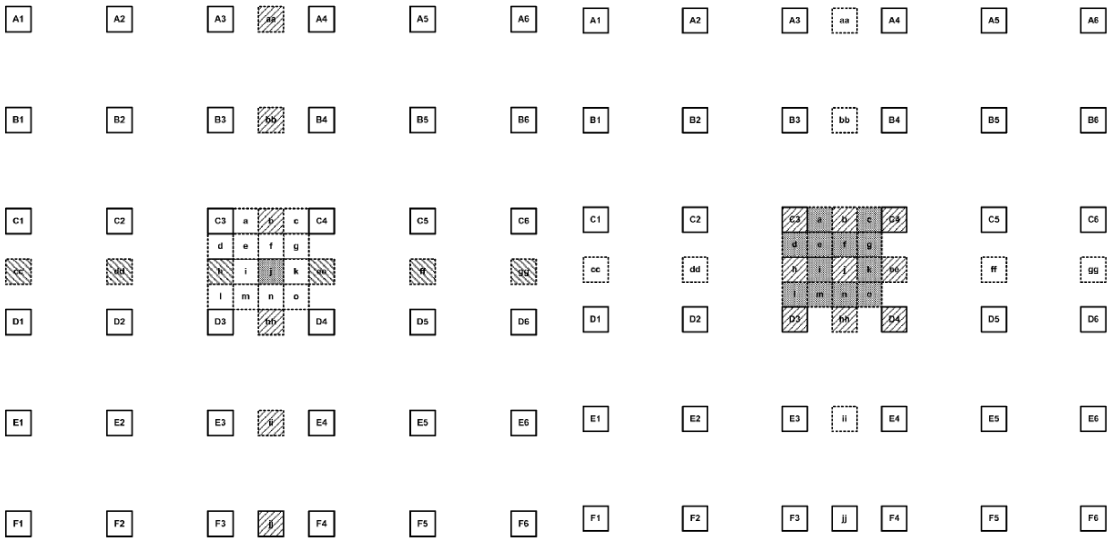


FIG. 4C

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 2006, p. 141
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 142
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 143
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures of limitations 1[b] and 1[c], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 I, 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. 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. *Supra claim 2.* *Supra limitation 1[c].* *See, e.g.,* Karczewicz I, ¶0007: > For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to

Nokia Exhibit 2006, p. 144
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0053: 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 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. For demonstrative purposes, the techniques described in this disclosure are described as being performed by motion compensation unit 35. Karczewicz I, ¶0054: Motion estimation unit 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of one or more predictive coded units (e.g., a previous and/or future frame in terms of time or temporally). Motion estimation unit 32 may, as an example, select a motion vector for a B-frame in a number of ways. In one way, motion estimation unit 32 may select a previous or future frame from a first set of frames (referred to

Nokia Exhibit 2006, p. 145
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	as list 0) and determine a motion vector using only this previous or future frame from list 0. Alternatively, motion estimation unit 32 may select a previous or future frame from a second set of frames (referred to as list 1) and determine a motion vector using only this previous or future frame from list 1. In yet another way, motion estimation unit 32 may select a first frame from list 0 and a second frame from list 1 and select one or more motion vectors from the first frame of list 0 and the second frame of list 1. This third form of prediction may be referred to as bi-predictive motion estimation. Techniques of this disclosure may be implemented so as to efficiently select a motion-compensated bi-prediction mode. The selected motion vector for any given list may point to a predictive video block that is most similar to the video block being coded, e.g., as defined by a metric such as sum of absolute difference (SAD) or sum of squared difference (SSD) of pixel values of the predictive block relative to pixel values of the block being coded. Karczewicz II, ¶0004: 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. Karczewicz II, ¶0056:

Nokia Exhibit 2006, p. 146
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 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. Karczewicz II, ¶0073: 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. Karczewicz II, ¶0074: Again, FIGS. 4A-4D show the integer-pixel samples (also called full-pixels) in the solid boxes with upper-case letters. For any given integer-pixel sample, there are altogether 15 sub-pixel positions, which are shown for integer-pixel sample "C3" and labeled "a" through "o" in FIGS. 4A-4D. In H.264/AVC, the half-pixel positions "b," "h," and "j" may be calculated first using

Nokia Exhibit 2006, p. 147
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	the one-dimensional 6-tap Wiener filter. As noted, half-pixel position "j" may require two levels of this interpolation using 6-tap filtering. In this case, for pixel position "j," the interpolation filter is applied first in the horizontal direction and then in the vertical direction, possibly with intermediate rounding of interpolated pixel values after interpolation in one dimension in order to ensure that data can be stored within sixteen-bit storage elements. Remaining quarter-pixel positions are then interpolated via filtering that uses a bilinear filter and the already calculated half-pixel samples. Karczewicz II, ¶0096: 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. 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

Nokia Exhibit 2006, p. 148
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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.

Karczewicz II, ¶0093:

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:
b1=C1−5*C2+20*C3+20*C4−5*C5+C6
where "C1," "C2," "C3," "C4," "C5" and "C6" represent the six closest integer pixels that

Nokia Exhibit 2006, p. 149
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. h1=A3−5*B3+20*C3+20*D3−5*E3+F3 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. See also the disclosures of limitations 1[b] and 1[c], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. To the extent this limitation is not disclosed by the combination of Karczewicz I and Karczewicz II, it would have been obvious to obtain the second prediction by shifting values of the second reference block to the left. Left-shifting to multiply a value by a coefficient and increase the precision of intermediate values in calculations was well known in the art. *See, e.g.*, U.S. Patent App. Pub. 2008/0198935 ("Srinivasan") at ¶91, ¶116, ¶124, ¶129, ¶131; U.S. Patent App. Pub. No. 2013/0034158 ("Kirchhoffer-158") at ¶67, ¶¶84-85, ¶103. This confirms that it would have been obvious to shift values of the second reference block to the left. Karczewicz I likewise teaches that bi-prediction can be performed from two different lists of data, thus acknowledging that the first prediction and the second prediction can be obtained using different methods. Among other reasons, a PHOSITA would have found it obvious to obtain the second prediction by left shifting and would have been motivated to do so because, in computing terms, left-shifting is a simpler operation than multiplication and can be performed without the need for a multiplication unit. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

Nokia Exhibit 2006, p. 150
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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 I, 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. *Supra claim 1.* *Supra limitations 1[d] and 1[e].* Karczewicz I, ¶0060: Default weighted prediction may be defined by the following equations for unidirectional prediction and bidirectional prediction, respectively. Unidirectional prediction: pred(i,j)=pred0(i,j) Bidirectional prediction: pred(i,j)=(pred0(i,j)+pred1(i,j)+1)>>1 where pred0(i,j) and pred1(i,j) are prediction data from list 0 and list 1. Karczewicz I, ¶0055: According to the ITU-T H.264/AVC standard, three motion-compensated bi-predictive algorithms or modes may be used to predict a B-frame or portions thereof, such as video blocks, macroblocks or any other discreet and/or contiguous portion of a B-frame. A first motion-compensated bi-predictive algorithm or mode, which is commonly referred to as default weighted prediction, may involve applying default weights to each identified video block of the first frame of list 0 and the second frame of list 1. The default weights may be programmed according to the standard, and are often selected to be equal for default weighted prediction. The weighted blocks of the first and second frames are then added together and divided by the

Nokia Exhibit 2006, p. 151
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	total number of frames used to predict the B-frame, e.g., two in this instance. Often, this division is accomplished by adding 1 to the addition of the weighted blocks of the first and second frames and then shifting the result to the right by one bit. The addition of 1 is a rounding adjustment. Karczewicz I, ¶0063: Conventionally, rounding adjustments are always used in bidirectional prediction. According to the equations above, a rounding adjustment of 1 is used in the default weighted prediction prior to a right shift by one, and a rounding adjustment of 32 is used in the implicit weighted prediction prior to a right shift by six. Generally, a rounding adjustment of 2r−1 is commonly used prior to a right shift by r, where r represents a positive integer. 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. *See, e.g.,* Karczewicz II, ¶¶0096-0106: 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 2006, p. 152
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 153
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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.

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

Nokia Exhibit 2006, p. 154
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 155
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 156
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 157
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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

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 2006, p. 158
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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

Nokia Exhibit 2006, p. 159
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	rounding in pixel location "j" will not impact the accuracy of the final value of quarter-pixels which rely on "j." See also the disclosures of limitations 1[d] and 1[e], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. As evidenced by the above disclosures, a POSITA would have understood that the equations set forth above for each of Scenarios 1–3 to comprise a rounding offset. Additionally, it would have been obvious for said decreasing to include the claimed rounding offset because the insertion of the rounding offset is performed right before and in conjunction with the right-shifting to affect the direction of the rounding. Karczewicz I and II include a rounding offset to control rounding error resulting from the right-shift operation that decreases precision. This was common in the art. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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	Karczewicz I, 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. 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.

Nokia Exhibit 2006, p. 160
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
prediction and values of said second prediction.	*See limitation 1[a].* *Supra limitations 1[b]-1[c].* *Supra claim 1.* *See, e.g.,* Karczewicz II, ¶0089: > 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). 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

Nokia Exhibit 2006, p. 161
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 162
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 163
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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."

Nokia Exhibit 2006, p. 164
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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 2006, p. 165
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures of limitations 1[b], 1[c], and 1[d], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 I, as evidenced by the example citations below, discloses an apparatus for encoding a block of pixels. *See, e.g.*, *See limitation 1[preamble]*. Karczewicz I, ¶0003: > Digital multimedia capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless communication devices, 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, digital media players, and the like. Digital multimedia devices may implement video coding techniques, such as MPEG-2, ITU-H.263, MPEG-4, or ITU-H.264/MPEG-4 Part 10, Advanced Video Coding (AVC), to transmit and receive or store and retrieve digital video data more efficiently. Video encoding techniques may perform video

Nokia Exhibit 2006, p. 166
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	compression via spatial and temporal prediction to reduce or remove redundancy inherent in video sequences. Karczewicz I, ¶0010: In another example, this disclosure describes a video encoder apparatus that encodes video data, the video encoder apparatus comprising a memory that stores the video data, and a motion compensation unit that selects between default weighted prediction and implicit weighted prediction, upon selecting default weighted prediction, selects between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the default weighted prediction, upon selecting implicit weighted prediction, selects between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the implicit weighted prediction, wherein the video encoder apparatus encodes the video data using weighed prediction data defined by the selections. Karczewicz I, ¶0012: 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. Karczewicz I, ¶0038: Video encoder 22 and video decoder 28 each may be implemented as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits

Nokia Exhibit 2006, p. 167
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	(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 encoder/decoder (CODEC) in a respective mobile device, subscriber device, broadcast device, server, or the like. Karczewicz I, ¶0098: The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units. The techniques described herein may also be implemented in hardware, software, firmware, or any combination thereof. Any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. In some cases, various features may be implemented as an integrated circuit device, such as an integrated circuit chip or chipset. Karczewicz I, ¶0099: If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may

Nokia Exhibit 2006, p. 168
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. Karczewicz I, ¶0100: The code or instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video codec. Also, the techniques could be fully implemented in one or more circuits or logic elements. Karczewicz II, as evidenced by the example citations below, discloses an apparatus for encoding a block of pixels. *See, e.g.,* *See limitation 1[preamble].* Karczewicz II, ¶0003: 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

Nokia Exhibit 2006, p. 169
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0015: 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 data based on a plurality of different pre-defined interpolation filters, and the video encoder encodes the video data based on the prediction data. Karczewicz II, ¶0017: 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. Karczewicz II, ¶0021: 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.

Nokia Exhibit 2006, p. 170
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0023: 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 encoded by the motion compensation unit to another device. Karczewicz II, ¶0025: 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. Karczewicz II, ¶0026: 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. Karczewicz II, ¶0048:

Nokia Exhibit 2006, p. 171
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. See also the disclosures of limitations 1[preamble], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 I, as evidenced by the example citations below, discloses at least one processor and at least one memory including computer program code. *See, e.g.,* Karczewicz I, ¶0003: Digital multimedia capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless communication devices, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, digital cameras, digital recording devices, video gaming devices, video game consoles, cellular or

Nokia Exhibit 2006, p. 172
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	satellite radio telephones, digital media players, and the like. Digital multimedia devices may implement video coding techniques, such as MPEG-2, ITU-H.263, MPEG-4, or ITU-H.264/MPEG-4 Part 10, Advanced Video Coding (AVC), to transmit and receive or store and retrieve digital video data more efficiently. Video encoding techniques may perform video compression via spatial and temporal prediction to reduce or remove redundancy inherent in video sequences. Karczewicz I, ¶0005: For inter-coding, a video encoder performs motion estimation to track the movement of matching video blocks between two or more adjacent frames or other coded units, such as slices of frames. Inter-coded frames may include predictive frames ("P-frames"), which may include blocks predicted from a previous frame, and bidirectional predictive frames ("B-frames"), which may include blocks predicted from a previous frame and a subsequent frame of a video sequence. The terms P-frames and B-frames are somewhat historic in the sense that early coding techniques limited prediction in specific directions. Newer coding formats and standards may not limit the prediction direction of P-frames or B-frames. Thus, the term "bi-directional" now refers to prediction based on two or more lists of reference data regardless of the temporal relationship of such reference data relative to the data being coded. Karczewicz I, ¶0006: Consistent with newer video standards such as ITU H.264, for example, bi-directional prediction may be based on two different lists which do not necessarily need to have data that resides temporally before and after the current video block. In other words, B-video blocks may be predicted from two lists of data, which may correspond to data from two previous frames, two subsequent frames, or one previous frame and one subsequent frame. In contrast, P-video blocks are predicted based on one list, i.e., one data structure, which may correspond to one predictive frame, e.g., one previous frame or one subsequent frame. B-frames and P-frames

Nokia Exhibit 2006, p. 173
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	may be more generally referred to as P-units and B-units. P-units and B-units may also be realized in smaller coded units, such as slices of frames or portions of frames. B-units may include B-video blocks, P-video blocks or I-video blocks. P-units may include P-video blocks or I-video blocks. I-units may include only I-video blocks. Karczewicz I, ¶0007: For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0012: 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. Karczewicz I, ¶0022:

Nokia Exhibit 2006, p. 174
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block is predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, rounding adjustments to bi-directional predictive data may be purposely eliminated to provide predictive data that lacks any rounding bias. In this case, rounded and unrounded predictive data may both be considered in a rate-distortion analysis to identify the best data for prediction of a given video block. One or more syntax elements may be encoded to indicate the selection, and a decoder may interpret the one or more syntax elements in order to determine whether rounding should be used in the decoding process. Karczewicz I, ¶0029: FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite radiotelephones, or any devices that can communicate video information over a communication channel 15, which may or may not be wireless. The techniques of this disclosure, however, which concern non-zero rounding and prediction mode selection techniques, are not necessarily limited to wireless applications or settings. Karczewicz I, ¶0030: 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 non-zero rounding

Nokia Exhibit 2006, p. 175
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	and prediction mode selection techniques as part of a video encoding process. Video decoder 28 may receive one or more syntax elements indicating the selection and indicating whether non-zero rounding was used. Accordingly, video decoder 28 may perform the proper weighted prediction signaled in the received bistream. Karczewicz I, ¶0032: Video source 20 of source device 12 may include a video capture device, such as a video camera, a video archive containing previously captured video, or a video feed from a video content provider. As a further alternative, video source 20 may generate computer graphics-based data as the source video, or a combination of live video, archived video, and computer-generated video. In some cases, if video source 20 is a video camera, source device 12 and destination device 16 may form so-called camera phones or video phones. In each case, the captured, pre-captured or computer-generated video may be encoded by video encoder 22. The encoded video information may then be modulated by modem 23 according to a communication standard, e.g., such as code division multiple access (CDMA) or another communication standard, and transmitted to destination device 16 via transmitter 24. Modem 23 may include various mixers, filters, amplifiers or other components designed for signal modulation. Transmitter 24 may include circuits designed for transmitting data, including amplifiers, filters, and one or more antennas. Karczewicz I, ¶0038: 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

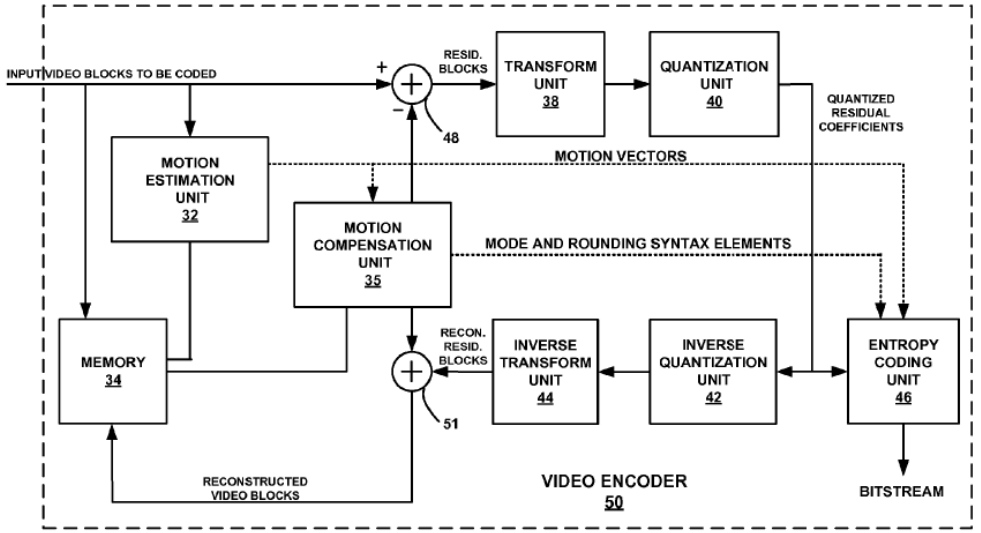
Nokia Exhibit 2006, p. 176
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	combined encoder/decoder (CODEC) in a respective mobile device, subscriber device, broadcast device, server, or the like. Karczewicz I, ¶0039: A video sequence typically includes a series of video frames. Video encoder 22 and video decoder 28 may operate on video blocks within individual video frames in order to encode and decode the video data. The video blocks may have fixed or varying sizes, and may differ in size according to a specified coding standard. Each video frame may include a series of slices or other independently decodable units. Each slice may include a series of macroblocks, which may be arranged into sub-blocks. As an example, the ITU-T 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. Karczewicz I, ¶0040: 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 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 macroblocks and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Alternatively, frames themselves may be decodable units, or other portions of a frame may be defined as decodable units. The term "coded unit" refers to any independently decodable unit of a video frame such as an entire frame, a slice of a frame,

Nokia Exhibit 2006, p. 177
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	a group of pictures (GOPs), or another independently decodable unit defined according to the coding techniques used. Karczewicz I, ¶0042: The techniques of this disclosure are specifically applicable to weighted bi-directional prediction. As mentioned above, bi-directional prediction is prediction of so-called "B-video blocks" based on two different lists of data. B-video blocks may be predicted from two lists of data from two previous frames, two lists of data from subsequent frames, or one list of data from a previous frame and one from a subsequent frame. In contrast, P-video blocks are predicted based on one list, which may correspond to one predictive frame, e.g., one previous frame or one subsequent frame. B-frames and P-frames may be more generally referred to as P-units and B-units. P-units and B-units may also be realized in smaller coded units, such as slices of frames or portions of frames. B-units may include B-video blocks, P-video blocks or I-video blocks. P-units may include P-video blocks or I-video blocks. I-units may include only I-video blocks. Karczewicz I, ¶0007: For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units.

Nokia Exhibit 2006, p. 178
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, Fig. 2:  FIG. 2 Karczewicz I, ¶0053: 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

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. For demonstrative purposes, the techniques described in this disclosure are described as being performed by motion compensation unit 35. Karczewicz I, ¶0054: Motion estimation unit 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of one or more predictive coded units (e.g., a previous and/or future frame in terms of time or temporally). Motion estimation unit 32 may, as an example, select a motion vector for a B-frame in a number of ways. In one way, motion estimation unit 32 may select a previous or future frame from a first set of frames (referred to as list 0) and determine a motion vector using only this previous or future frame from list 0. Alternatively, motion estimation unit 32 may select a previous or future frame from a second set of frames (referred to as list 1) and determine a motion vector using only this previous or future frame from list 1. In yet another way, motion estimation unit 32 may select a first frame from list 0 and a second frame from list 1 and select one or more motion vectors from the first frame of list 0 and the second frame of list 1. This third form of prediction may be referred to as bi-predictive motion estimation. Techniques of this disclosure may be implemented so as to efficiently select a motion-compensated bi-prediction mode. The selected motion vector for any given list may point to a predictive video block that is most similar to the video block being coded, e.g., as defined by a metric such as sum of absolute difference (SAD) or sum of squared difference (SSD) of pixel values of the predictive block relative to pixel values of the block being coded. Karczewicz I, ¶¶0051-76, reproduced *supra* for claim 1.

Nokia Exhibit 2006, p. 180
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, Fig. 1 and 2, reproduced *supra*. Karczewicz I, ¶¶0098-100: [0098] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units. The techniques described herein may also be implemented in hardware, software, firmware, or any combination thereof. Any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. In some cases, various features may be implemented as an integrated circuit device, such as an integrated circuit chip or chipset. [0099] If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. [0100] The code or instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific

Nokia Exhibit 2006, p. 181
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video codec. Also, the techniques could be fully implemented in one or more circuits or logic elements. Karczewicz II, as evidenced by the example citations below, discloses at least one processor and at least one memory including computer program code. *See, e.g.,* Karczewicz II, ¶0003: 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. Karczewicz II, ¶0004: 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

Nokia Exhibit 2006, p. 182
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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.

Karczewicz II, Fig. 2:

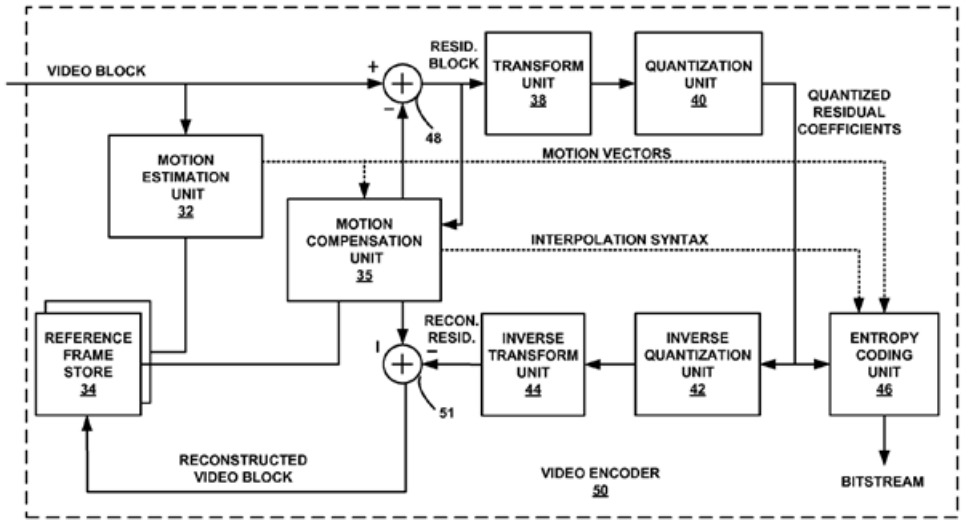


FIG. 2

Nokia Exhibit 2006, p. 183
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0025: 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. Karczewicz II, ¶0026: 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. Karczewicz II, ¶0027: 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. Karczewicz II, ¶0040:

Nokia Exhibit 2006, p. 184
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite radiotelephones, or any wireless devices that can communicate video information over a communication channel 15, in which case communication channel 15 is wireless. The techniques of this disclosure, however, which concern motion compensated interpolation, are not necessarily limited to wireless applications or settings. Karczewicz II, ¶0041: 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. Karczewicz II, ¶0043: Video source 20 of source device 12 may include a video capture device, such as a video camera, a video archive containing previously captured video, or a video feed from a video content provider. As a further alternative, video source 20 may generate computer graphics-based data as the source video, or a combination of live video, archived video, and computer-generated video. In some cases, if video source 20 is a video camera, source device 12 and destination device 16 may form so-called camera phones or video phones. In each case, the

Nokia Exhibit 2006, p. 185
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	captured, pre-captured or computer-generated video may be encoded by video encoder 22. The encoded video information may then be modulated by modem 23 according to a communication standard, e.g., such as code division multiple access (CDMA) or another communication standard, and transmitted to destination device 16 via transmitter 24. Modem 23 may include various mixers, filters, amplifiers or other components designed for signal modulation. For wireless applications, transmitter 24 may include circuits designed for transmitting data, including amplifiers, filters, and one or more antennas. Karczewicz II, ¶0048: 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. 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 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

Nokia Exhibit 2006, p. 186
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	of transformation coefficients, e.g., following a transformation process such as discrete cosine transform or a conceptually similar transformation process. Karczewicz II, ¶0050: 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. Karczewicz II, ¶0054: FIG. 2 is a block diagram illustrating an example of a video encoder 50 that may perform motion compensated interpolation consistent with this disclosure. Video encoder 50 may correspond to video encoder 22 of device 20, or a video encoder of a different device. Video encoder 50 may perform intra- and inter-coding of blocks within video frames, although intra-coding is not illustrated. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames of a video sequence. Intra-mode (I-mode) may refer to the spatial based compression mode and Inter-modes such as prediction (P-mode) or bi-directional (B-mode) may refer to the temporal based compression modes. The techniques of this disclosure typically may be applied during inter-coding, and therefore, intra-coding units such as spatial prediction unit are not illustrated in FIG. 2 for

Nokia Exhibit 2006, p. 187
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	simplicity and ease of illustration. However, the rounding techniques of this disclosure may also be applicable to spatial prediction and intra coding techniques. Karczewicz II, ¶0055: As shown in FIG. 2, video encoder 50 receives a current video block within a video frame to be encoded. In the example of FIG. 2, video encoder 50 includes motion estimation unit 32, a motion compensation unit 35, a reference frame store 34, an adder 48, a transform unit 38, a quantization unit 40, and an entropy coding unit 46. For video block reconstruction, video encoder 50 also includes an inverse quantization unit 42, an inverse transform unit 44 adder 51. A deblocking filter (not shown) may also be included to filter block boundaries to remove blockiness artifacts from reconstructed video. If desired, the deblocking filter would typically filter the output of adder 51. 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 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

Nokia Exhibit 2006, p. 188
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	motion compensation unit 35. However, interpolation may be performed during motion estimation in order to facilitate the selection of the best motion vectors. Karczewicz II, ¶0057: 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. Karczewicz II, ¶0058: 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 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

Nokia Exhibit 2006, p. 189
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	interpolation filter, and video encoder applies adder 48 to subtract that predictive data from the video block being coded to generate residual data. Karczewicz II, ¶0059: 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. Karczewicz II, ¶0060: 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.

Nokia Exhibit 2006, p. 190
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	The transform may convert the residual information from a pixel domain to a frequency domain. Karczewicz II, ¶0061: Quantization unit 40 quantizes the residual transform coefficients to further reduce bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, a 16-bit value may be rounded down to a 15-bit value during quantization. Following quantization, entropy coding unit 46 entropy codes the quantized transform coefficients. For example, entropy coding unit 46 may perform content adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), or another entropy coding methodology. Following the entropy coding by entropy coding unit 46, the encoded video may be transmitted to another device or archived for later transmission or retrieval. The coded bitstream may include entropy coded residual blocks, motion vectors for such blocks, and other syntax including the interpolation syntax that identifies the interpolation filters that were applied by motion compensation unit 35. Karczewicz II, ¶0062: Inverse quantization unit 42 and inverse transform unit 44 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual block in the pixel domain, e.g., for later use a reference block. Summer 51 adds the reconstructed residual block to the motion compensated prediction block produced by motion compensation unit 35 to produce a reconstructed video block for storage in reference frame store 34. The reconstructed video block may be used by motion estimation unit 32 and motion compensation unit 35 as a reference block to inter-encode a block in a subsequent video frame. Karczewicz II, ¶0073:

Nokia Exhibit 2006, p. 191
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0089: 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). 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

Nokia Exhibit 2006, p. 192
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 2006, p. 193
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

TABLE 5

positions {a, c, d, 1} 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 2006, p. 194
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

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, ¶0118-120:

> [0118] The techniques of this disclosure may be embodied in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units, etc.
>
> [0119] Accordingly, the techniques described herein may be implemented in hardware, software, firmware, or any combination thereof. If implemented in hardware, any features

Nokia Exhibit 2006, p. 195
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. [0120] The code may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video encoder-decoder (CODEC). Also, the techniques could be fully implemented in one or more circuits or logic elements. Based on these disclosures, a POSITA would have understood that, in a conventional computing device, the processor executes computer program code in the memory which is configured to cause the device to carry out its functionalities. Therefore, Karczewicz I and II teach an encoder and decoder device as recited in in this limitation that performs operations as described in the remaining limitations of this claim. Alternatively, it would have been obvious to a POSITA to implement the teachings of

Nokia Exhibit 2006, p. 196
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I and II in the claimed processor and memory because that is a conventional computing architecture for performing encoding operations. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 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;	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See limitation 1[a].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 197
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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;	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See limitation 1[b].* *Supra limitation 7[b].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[d] use said second reference block to obtain a second prediction, said second prediction having the second precision;	The combination of Karczewicz I and Karczewicz II discloses using the second reference block to obtain a second prediction, the second prediction having the second precision. *See, e.g.,* *See limitation 1[c].* *Supra limitations 7[b]-7[c].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*.

Nokia Exhibit 2006, p. 198
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[e] obtain a combined prediction based at least partly upon said first prediction and said second prediction;	The combination of Karczewicz I and Karczewicz II discloses a combined prediction based at least partly upon the first prediction and the second prediction. *See, e.g.*, *See limitation 1[d]*. *Supra limitations 7[b]-7[d]*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and	The combination of Karczewicz I and Karczewicz II discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right. *See, e.g.*,

Nokia Exhibit 2006, p. 199
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See limitation 1[e]. *Supra limitations 7[d]-7[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See limitation 1[f].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 200
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 2.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
9. The apparatus according to claim 8, wherein said first prediction is obtained by interpolation using values of said first reference block by: right shifting a sum of a P-tap filter using values of said first reference block.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 3*. *Supra* claim 1. *See* claim 2.

Nokia Exhibit 2006, p. 201
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 4.* *Supra claim 2.* *Supra limitation 1[c].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 202
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 5.* *Supra claim 1.* *Supra limitations 1[d] and 1[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 6.*

Nokia Exhibit 2006, p. 203
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
prediction and values of said second prediction.	*Supra limitations 1[b]-1[c].* *Supra claim 1.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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:	The combination of Karczewicz I and Karczewicz II discloses 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. *See, e.g.,* *See limitations 1[preamble], 7[preamble], and 7[a].* *See, e.g.,* Karczewicz I, ¶0012: > 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

Nokia Exhibit 2006, p. 204
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	software that executes the techniques may be initially stored in a computer-readable medium and loaded and executed in the processor. Karczewicz I, ¶0013: Accordingly, this disclosure also contemplates a computer-readable storage medium comprising instructions that when executed cause a processor to select between default weighted prediction and implicit weighted prediction, upon selecting default weighted prediction, select between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the default weighted prediction, upon selecting implicit weighted prediction, select between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the implicit weighted prediction, and encode the video data using weighed prediction data defined by the selections. Karczewicz I, ¶0038: 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 encoder/decoder (CODEC) in a respective mobile device, subscriber device, broadcast device, server, or the like. Karczewicz I, ¶¶0098-100: [0098] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip

Nokia Exhibit 2006, p. 205
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units. The techniques described herein may also be implemented in hardware, software, firmware, or any combination thereof. Any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. In some cases, various features may be implemented as an integrated circuit device, such as an integrated circuit chip or chipset. [0099] If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. [0100] The code or instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for

Nokia Exhibit 2006, p. 206
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	encoding and decoding, or incorporated in a combined video codec. Also, the techniques could be fully implemented in one or more circuits or logic elements. *See, e.g.*, Karczewicz II, ¶0025: 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. Karczewicz II, ¶0026: 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. Karczewicz II, ¶0028: 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

Nokia Exhibit 2006, p. 207
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0048: 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. Karczewicz II, ¶0118-120: [0118] The techniques of this disclosure may be embodied in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units, etc. [0119] Accordingly, the techniques described herein may be implemented in hardware, software, firmware, or any combination thereof. If implemented in hardware, any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as

Nokia Exhibit 2006, p. 208
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. [0120] The code may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video encoder-decoder (CODEC). Also, the techniques could be fully implemented in one or more circuits or logic elements. Based on these disclosures, Karczewicz I and Karczewicz II implement their teachings using program code, executable by a processor, residing on computer-readable storage media, including memory types, hard disks, and CD-ROMs, which were forms of non-transitory media as opposed to transitory signals. Software modules obviously comprise executable program code portions including program code instructions. Karczewicz I and Karczewicz II apply their teachings to a computer-program product. The operations taught by Karczewicz I and Karczewicz II, performed by the software module, teach the limitations of claim 13. Additionally, a POSITA would have found it obvious to have a computer program product that includes a storage medium storing the software module, since computer program products have almost universally stored their software program code in non-transitory mediums for decades. Therefore, Karczewicz I and Karczewicz II teach a computer program product as recited in claim 13 that performs the operations recited in claim 13.

Nokia Exhibit 2006, p. 209
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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;	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See limitation 1[a].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
13[b] use said first reference block to obtain a first prediction, said first prediction	The combination of Karczewicz I and Karczewicz II 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.

Nokia Exhibit 2006, p. 210
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
having a second precision, which is higher than said first precision;	*See, e.g.*, *See limitation 1[b]*. *Supra limitation 1[preamble].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
13[c] use said second reference block to obtain a second prediction, said second prediction having the second precision;	The combination of Karczewicz I and Karczewicz II discloses using the second reference block to obtain a second prediction, the second prediction having the second precision. *See, e.g.*, *See limitation 1[c]*. *Supra limitation 13[b].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

Nokia Exhibit 2006, p. 211
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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.
13[d] obtain a combined prediction based at least partly upon said first prediction and said second prediction;	The combination of Karczewicz I and Karczewicz II discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction. *See, e.g.,* *See limitation 1[d].* *Supra limitations 13[b]-13[c].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
13[e] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and	The combination of Karczewicz I and Karczewicz II discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right. *See, e.g.,* *See limitation 1[e].*

Nokia Exhibit 2006, p. 212
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Supra limitations 13[b]-13[d]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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 pixels.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See limitation 1[f].* *Supra limitations 13[d]-13[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 213
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 2.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See* claim 3. *Supra claim 2.* *Supra limitation 1[c].*

Nokia Exhibit 2006, p. 214
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
16. The computer program product according to claim 14, wherein in an instance in which said second motion vector points to an integer sample, said second prediction is obtained by shifting values of said second reference block to the left.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See* claim 4. *Supra claim 2.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
17. The computer program product according to claim 13,	The combination of Karczewicz I and Karczewicz II discloses that the program code instructions configured to decrease the precision of the combined prediction by shifting bits of the combined

Nokia Exhibit 2006, p. 215
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	prediction to the right, further comprise program code instructions configured to insert a rounding offset to the combined prediction before the decreasing. *See, e.g.,* *See claims 5 and 11.* *Supra claim 1.* *Supra limitations 1[d] and 1[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 6.* *See limitation 1[a].*

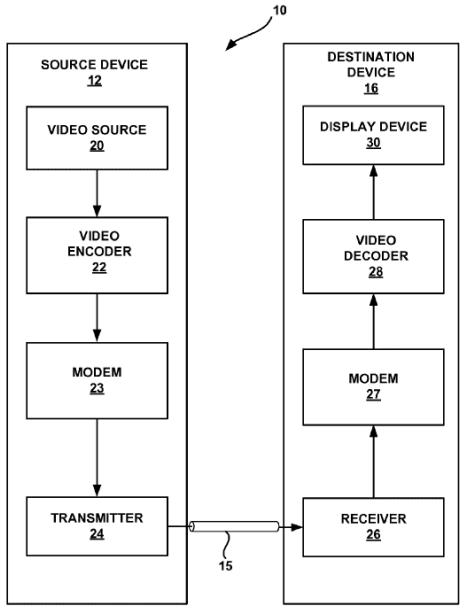
Nokia Exhibit 2006, p. 216
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
and values of said second prediction.	*Supra limitations 1[b]-1[c].* *Supra claim 1.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[preamble] A method for decoding a block of pixels, the method comprising:	Karczewicz I, as evidenced by the example citations below, discloses a method for decoding a block of pixels. *See, e.g.*, Karczewicz I, ¶0008: > This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block may be predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, techniques are described for selecting among default weighted prediction, implicit weighted prediction, and explicit weighted prediction. In this context, techniques are also described for adding offset to prediction data, e.g., using the format of explicit weighted prediction to allow for offsets to predictive data that is otherwise defined by implicit or default weighted prediction. Karczewicz I, ¶0009:

Nokia Exhibit 2006, p. 217
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	In one example, this disclosure describes a method of encoding video data, the method comprising selecting between default weighted prediction and implicit weighted prediction, upon selecting default weighted prediction, selecting between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the default weighted prediction, upon selecting implicit weighted prediction, selecting between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the implicit weighted prediction, and encoding the video data using weighed prediction data defined by the selections. Karczewicz I, ¶0022: This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block is predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, rounding adjustments to bi-directional predictive data may be purposely eliminated to provide predictive data that lacks any rounding bias. In this case, rounded and unrounded predictive data may both be considered in a rate-distortion analysis to identify the best data for prediction of a given video block. One or more syntax elements may be encoded to indicate the selection, and a decoder may interpret the one or more syntax elements in order to determine whether rounding should be used in the decoding process. Karczewicz I, Fig. 1:

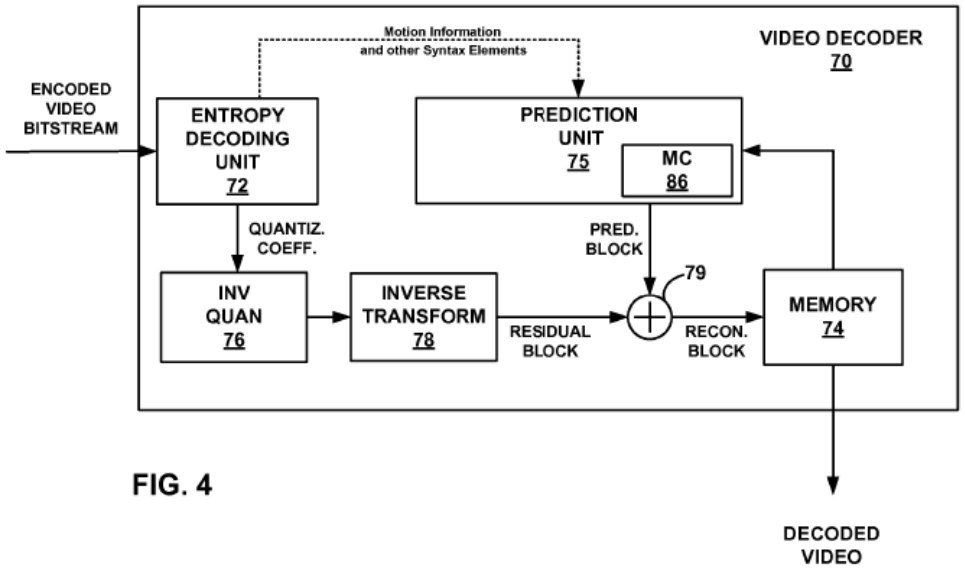
Nokia Exhibit 2006, p. 218
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 1 Karczewicz I, ¶0029: FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite

Nokia Exhibit 2006, p. 219
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	radiotelephones, or any devices that can communicate video information over a communication channel 15, which may or may not be wireless. The techniques of this disclosure, however, which concern non-zero rounding and prediction mode selection techniques, are not necessarily limited to wireless applications or settings. Karczewicz I, ¶0030: 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 non-zero rounding and prediction mode selection techniques as part of a video encoding process. Video decoder 28 may receive one or more syntax elements indicating the selection and indicating whether non-zero rounding was used. Accordingly, video decoder 28 may perform the proper weighted prediction signaled in the received bistream. Karczewicz I, ¶0033: Receiver 26 of destination device 16 receives information over channel 15, and modem 27 demodulates the information. Again, the video encoding process may implement one or more of the techniques described herein to provide non-zero rounding and prediction mode selection consistent with this disclosure. The information communicated over channel 15 may include information defined by video encoder 22, which may be used by video decoder 28 consistent with this disclosure. Display device 30 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a cathode ray tube, a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device.

Nokia Exhibit 2006, p. 220
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, Fig. 4:  FIG. 4 Karczewicz I, ¶0039: A video sequence typically includes a series of video frames. Video encoder 22 and video decoder 28 may operate on video blocks within individual video frames in order to encode and decode the video data. The video blocks may have fixed or varying sizes, and may differ in size according to a specified coding standard. Each video frame may include a series of slices or other independently decodable units. Each slice may include a series of macroblocks, which

Nokia Exhibit 2006, p. 221
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	may be arranged into sub-blocks. As an example, the ITU-T 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. Karczewicz I, ¶0040: 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 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 macroblocks and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Alternatively, frames themselves may be decodable units, or other portions of a frame may be defined as decodable units. The term "coded unit" refers to any independently decodable unit of a video frame such as an entire frame, a slice of a frame, a group of pictures (GOPs), or another independently decodable unit defined according to the coding techniques used. Karczewicz I, ¶0083-86: [0083] FIG. 4 is a block diagram illustrating an exemplary video decoder 70, which may perform the reciprocal decoding techniques to the encoding techniques described above. Video decoder 70 may include an entropy decoding unit 72, a prediction unit 75, an inverse quantization unit 76, an inverse transform unit 78, a memory 74 and adder 79. Prediction unit 75 may include a motion compensation (MC) unit 88, as well as spatial prediction components, which are not shown for simplicity and ease of illustration.

Nokia Exhibit 2006, p. 222
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0084] Video decoder 70 may receive encoded video data, and one or more syntax elements that indicate whether a rounding adjustment was used to encode the encoded video data. MC unit 86 of prediction unit 75 may generate weighted prediction data that depends on two or more lists of data, as described herein. In accordance with this disclosure, the weighted prediction data does not include the rounding adjustment if the one or more syntax elements indicate that the rounding adjustment was not used to encode the encoded video data. Video decoder 70 can decode the video data using the weighted prediction data, e.g., by invoking adder 79 to add the weighted prediction data (e.g., a prediction block) to residual data (e.g., a residual block). [0085] In general, entropy decoding unit 72 receives an encoded bitsteam and entropy decodes the bitstream to generate quantized coefficients, motion information and other syntax. The motion information (e.g., motion vectors) and other syntax are forwarded to prediction unit 75 for use in generating the predictive data. Prediction unit 75 performs bidirectional prediction consistent with this disclosure, avoiding rounding adjustments in some cases, and possibly implementing default, implicit or explicit weighted prediction according to the received syntax elements. The syntax elements may identify the type of weighted prediction that to be used, coefficients and offset if explicit weighted prediction to be used, and whether rounding adjustments should be used in the decoding. [0086] The quantized coefficients are sent from entropy decoding unit 72 to inverse quantization unit 76, which performs inverse quantization. Inverse transform unit 78 then inverse transforms the de-quantized coefficients back to the pixel domain to generate a residual block. Adder 79 combines the prediction data (e.g., a prediction block) generated by prediction unit 75 with the residual block from inverse transform unit 78 to create a reconstructed video block, which may be stored in memory 74 and/or output from video decoder 70 as decoded video output.

Nokia Exhibit 2006, p. 223
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, as evidenced by the example citations below, discloses a method for decoding a block of pixels. *See, e.g.,* Karczewicz II, Abstract: > 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. Karczewicz II, ¶0002: > This disclosure relates to digital video coding and, more particularly, fractional interpolations of predictive data used in video coding. Karczewicz II, ¶0006: > 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

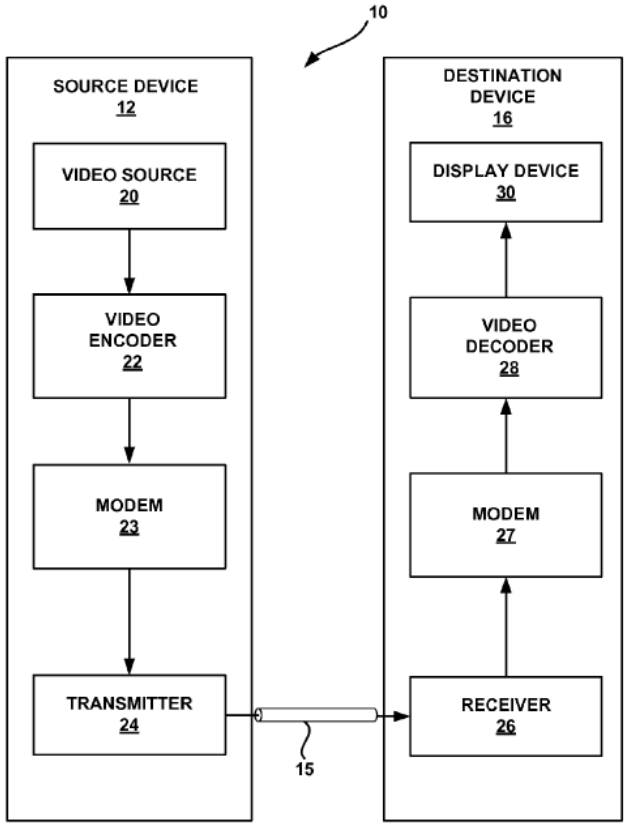
Nokia Exhibit 2006, p. 224
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0013: 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. Karczewicz II, ¶0014: 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. Karczewicz II, ¶0035: 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

Nokia Exhibit 2006, p. 225
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, Fig.1:

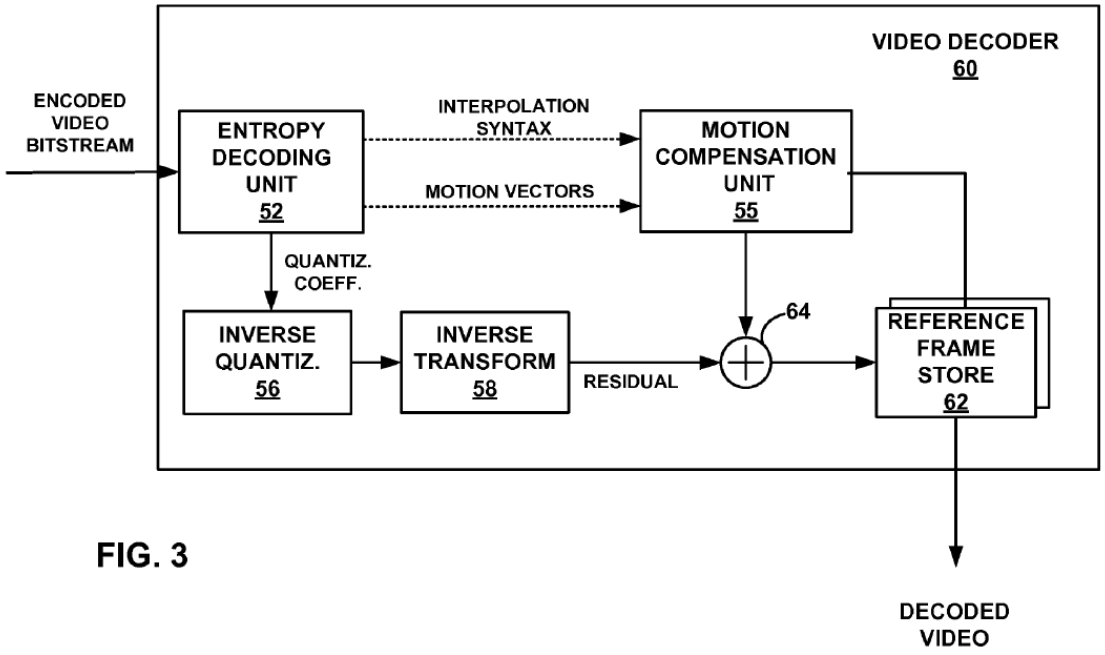
Nokia Exhibit 2006, p. 226
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 1 Karczewicz II, ¶0040-41:

Nokia Exhibit 2006, p. 227
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0040] FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite radiotelephones, or any wireless devices that can communicate video information over a communication channel 15, in which case communication channel 15 is wireless. The techniques of this disclosure, however, which concern motion compensated interpolation, are not necessarily limited to wireless applications or settings. [0041] 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. Karczewicz II, ¶0044: Receiver 26 of destination device 16 receives information over channel 15, and modem 27 demodulates the information. Again, the video encoding process may implement one or more of the techniques described herein to improve the interpolation during motion compensation. The video decoding process performed by video decoder 28 may also perform interpolation during its motion compensation stage of the decoding process. Display device 30 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a

Nokia Exhibit 2006, p. 228
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	cathode ray tube, a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device. Karczewicz II, Fig. 3:  FIG. 3 Karczewicz II, ¶0049:

Nokia Exhibit 2006, p. 229
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 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. Karczewicz II, ¶0050: 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. Karczewicz II, ¶0063-66: [0063] 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

Nokia Exhibit 2006, p. 230
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 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. [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 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). [0065] 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).

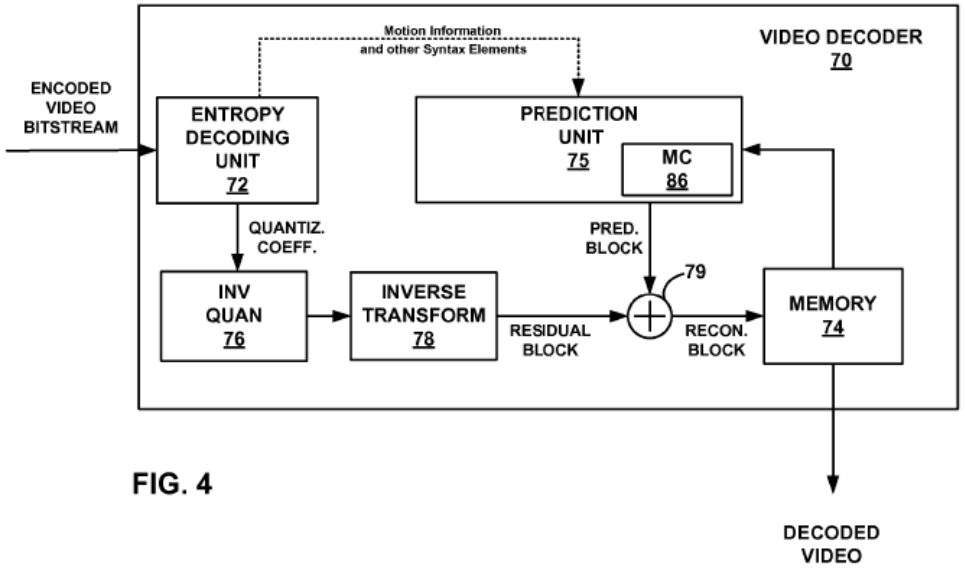
Nokia Exhibit 2006, p. 231
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0066] 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 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. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth supra. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 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 I, 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. *See, e.g.,* *See limitation 1[a], 19[preamble].* Karczewicz I, ¶0085: In general, entropy decoding unit 72 receives an encoded bitsteam and entropy decodes the bitstream to generate quantized coefficients, motion information and other syntax. The motion

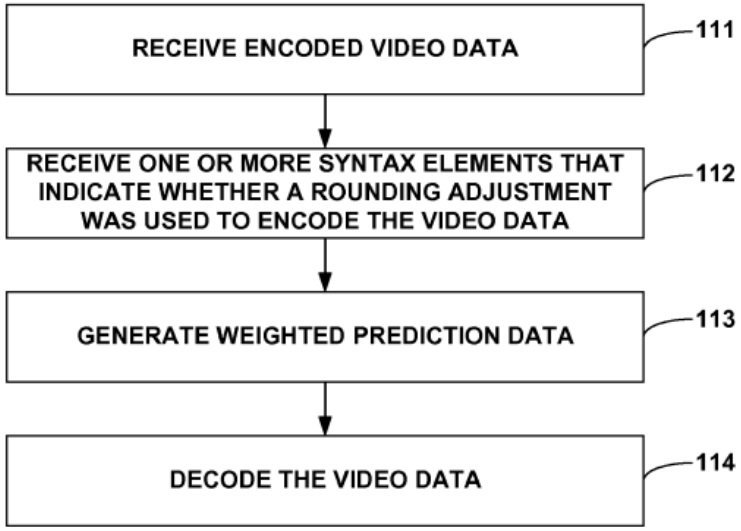
Nokia Exhibit 2006, p. 232
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	information (e.g., motion vectors) and other syntax are forwarded to prediction unit 75 for use in generating the predictive data. Prediction unit 75 performs bidirectional prediction consistent with this disclosure, avoiding rounding adjustments in some cases, and possibly implementing default, implicit or explicit weighted prediction according to the received syntax elements. The syntax elements may identify the type of weighted prediction that to be used, coefficients and offset if explicit weighted prediction to be used, and whether rounding adjustments should be used in the decoding. Karczewicz I, ¶0088-89: [0088] FIG. 6 is a flow chart illustrating an exemplary process performed by a video decoder consistent with this disclosure. FIG. 6 will be described from the perspective of video decoder 70 of FIG. 4. As shown in FIG. 6, video decoder receives encoded video data (111), and receives one or more syntax elements that indicate whether a rounding adjustment was used to encode the video data (1 12). In particular, entropy decoding unit 72 may receive an encoded bitstream that includes the video data and the one or more syntax elements. Following entropy decoding, entropy decoding unit 72 may output the video data as quantized transform coefficients, which are inverse quantized by unit 76 and inverse transformed by unit 78. Entropy decoding unit 72 may output syntax elements to prediction unit, which includes the one or more syntax elements that indicate whether a rounding adjustment was used to encode the video data, motion vectors and possibly other syntax. [0089] Prediction unit 75 invokes motion compensation unit 86 for block based predictive decoding. In doing so, motion compensation unit 86 generates weighted prediction data based on the syntax (113). Accordingly, if the one or more syntax elements indicate that a rounding adjustment was used, then motion compensation unit 86 generates weighted prediction data that includes the rounding adjustment. However, if the one or more syntax elements indicate that a rounding adjustment was not used, then motion compensation unit 86 generates weighted prediction data that lacks the rounding adjustment. Video decoder 70 can then decode the video

Nokia Exhibit 2006, p. 233
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	data using the weighted prediction data (114). In particular, video decoder 70 may invoke adder 79 to combine weighted prediction data (e.g., a prediction block) with residual video data (e.g., a residual block) in order to generate a reconstruction of the video data (e.g., a reconstructed video block). Karczewicz I, Fig. 4:  FIG. 4 Karczewicz I, Fig. 6:

Nokia Exhibit 2006, p. 234
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 6 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. *See, e.g.,* *See limitation 1[a].*

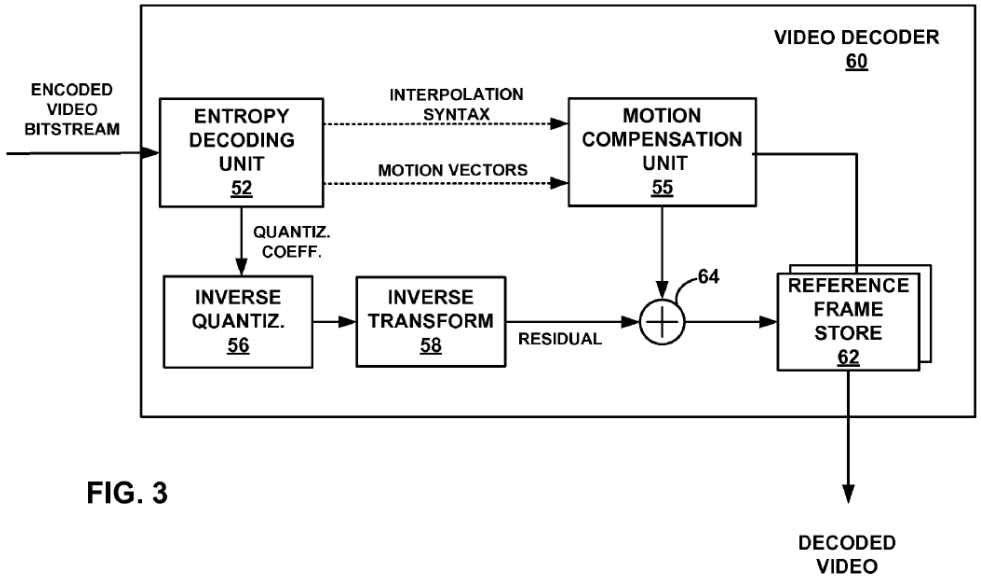
Nokia Exhibit 2006, p. 235
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0004: 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. Karczewicz II, ¶0006: 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.

Nokia Exhibit 2006, p. 236
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[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 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. Karczewicz II, ¶0063: 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 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

Nokia Exhibit 2006, p. 237
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	this interpolated prediction data, the video data (e.g., a reconstructed residual video block) can be decoded. Karczewicz II, Fig. 3:  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

Nokia Exhibit 2006, p. 238
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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). Karczewicz II, ¶0110: 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. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 239
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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 I, 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. *See, e.g.,* *See limitation 1[b], 19[preamble].* Karczewicz I, ¶0008: > This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block may be predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, techniques are described for selecting among default weighted prediction, implicit weighted prediction, and explicit weighted prediction. In this context, techniques are also described for adding offset to prediction data, e.g., using the format of explicit weighted prediction to allow for offsets to predictive data that is otherwise defined by implicit or default weighted prediction. Karczewicz I, ¶0022: > This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block is predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, rounding adjustments to bi-directional predictive data may be purposely eliminated to provide predictive data that lacks any rounding bias. In this case, rounded and unrounded predictive data may both be considered in a rate-distortion analysis to identify the best data for prediction of a given video block. One or more syntax elements may be encoded to indicate the selection, and a decoder may interpret the one or more syntax elements in order to determine whether rounding should be used in the decoding process.

Nokia Exhibit 2006, p. 240
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, ¶0031: The illustrated system 10 of FIG. 1 is merely exemplary. The non-zero rounding and prediction mode selection techniques of this disclosure may be performed by any coding device that supports bi-directional motion compensated prediction. Source device 12 and destination device 16 are merely examples of such coding devices in which source device 12 generates coded video data for transmission to destination device 16. In some cases, devices 12, 16 may operate in a substantially symmetrical manner such that, each of devices 12, 16 includes video encoding and decoding components. 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. Karczewicz I, ¶0035: 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). Karczewicz I, ¶0089: Prediction unit 75 invokes motion compensation unit 86 for block based predictive decoding. In doing so, motion compensation unit 86 generates weighted prediction data based on the syntax (113). Accordingly, if the one or more syntax elements indicate that a rounding

Nokia Exhibit 2006, p. 241
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	adjustment was used, then motion compensation unit 86 generates weighted prediction data that includes the rounding adjustment. However, if the one or more syntax elements indicate that a rounding adjustment was not used, then motion compensation unit 86 generates weighted prediction data that lacks the rounding adjustment. Video decoder 70 can then decode the video data using the weighted prediction data (114). In particular, video decoder 70 may invoke adder 79 to combine weighted prediction data (e.g., a prediction block) with residual video data (e.g., a residual block) in order to generate a reconstruction of the video data (e.g., a reconstructed video block). 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. *See, e.g.,* *See limitation 1[b].* Karczewicz II, ¶0008: 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

Nokia Exhibit 2006, p. 242
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0009: 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 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. Karczewicz II, ¶0013: 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.

Nokia Exhibit 2006, p. 243
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, ¶0016: 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. Karczewicz II, ¶0041: 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. Karczewicz II, ¶0042: 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

Nokia Exhibit 2006, p. 244
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	one-way or two-way video transmission between video devices 12, 16, e.g., for video streaming, video playback, video broadcasting, or video telephony. Karczewicz II, ¶0046: 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). Karczewicz II, ¶0063: 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). See also the disclosures of limitation 1[b], reproduced *supra*.

Nokia Exhibit 2006, p. 245
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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;	The combination of Karczewicz I and Karczewicz II discloses using the second reference block to obtain a second prediction, the second prediction having the second precision. *See, e.g.*, *See limitation 1[c], 19[preamble], 19[b].* See also the disclosures of limitation 1[b], reproduced *supra*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[d] obtaining a combined prediction based at least partly	The combination of Karczewicz I and Karczewicz II discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction.

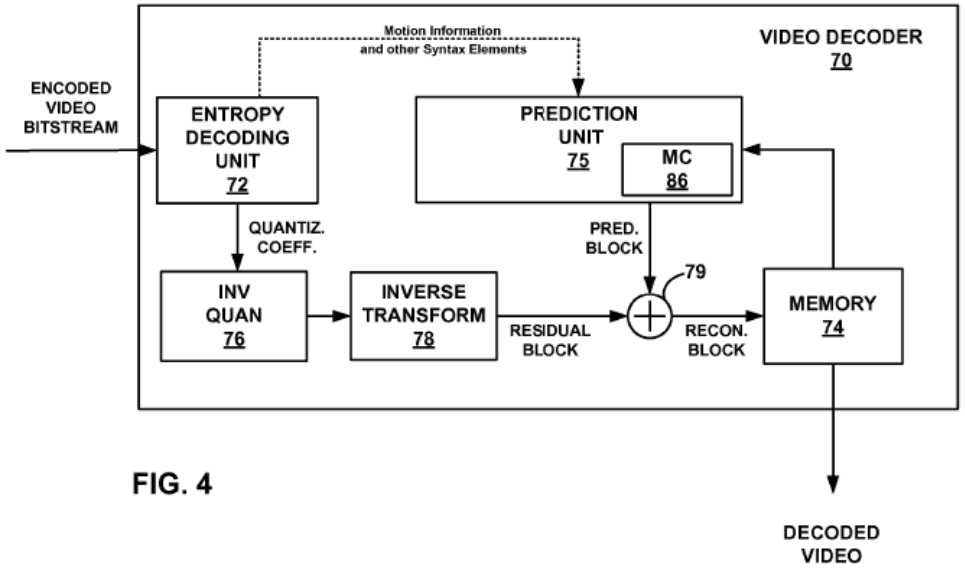
Nokia Exhibit 2006, p. 246
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
upon said first prediction and said second prediction;	*See, e.g.*, *See limitation 1[d]*. *Supra limitations 19[preamble], 19[b]-19[c]*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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	The combination of Karczewicz I and Karczewicz II discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right. *See, e.g.*, *See limitation 1[e]*. *Supra limitations 19[preamble], 19[b]-19[d]*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

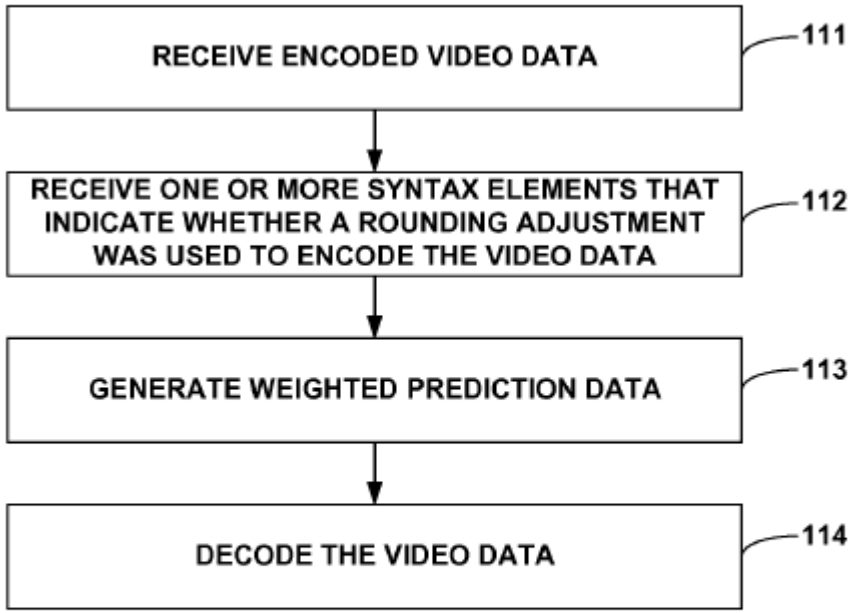
Nokia Exhibit 2006, p. 247
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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 I, as evidenced by the example citations below, discloses reconstructing the block of pixels based on the combined prediction. *See, e.g., Supra limitations 19[preamble], 19[b]-19[e].* Karczewicz I, ¶0086: > The quantized coefficients are sent from entropy decoding unit 72 to inverse quantization unit 76, which performs inverse quantization. Inverse transform unit 78 then inverse transforms the de-quantized coefficients back to the pixel domain to generate a residual block. Adder 79 combines the prediction data (e.g., a prediction block) generated by prediction unit 75 with the residual block from inverse transform unit 78 to create a reconstructed video block, which may be stored in memory 74 and/or output from video decoder 70 as decoded video output. Karczewicz I, ¶0089: > Prediction unit 75 invokes motion compensation unit 86 for block based predictive decoding. In doing so, motion compensation unit 86 generates weighted prediction data based on the syntax (113). Accordingly, if the one or more syntax elements indicate that a rounding adjustment was used, then motion compensation unit 86 generates weighted prediction data that includes the rounding adjustment. However, if the one or more syntax elements indicate that a rounding adjustment was not used, then motion compensation unit 86 generates weighted prediction data that lacks the rounding adjustment. Video decoder 70 can then decode the video data using the weighted prediction data (114). In particular, video decoder 70 may invoke adder 79 to combine weighted prediction data (e.g., a prediction block) with residual video data (e.g.,

Nokia Exhibit 2006, p. 248
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	a residual block) in order to generate a reconstruction of the video data (e.g., a reconstructed video block). Karczewicz I, Fig. 4:  FIG. 4 Karczewicz I, Fig. 6:

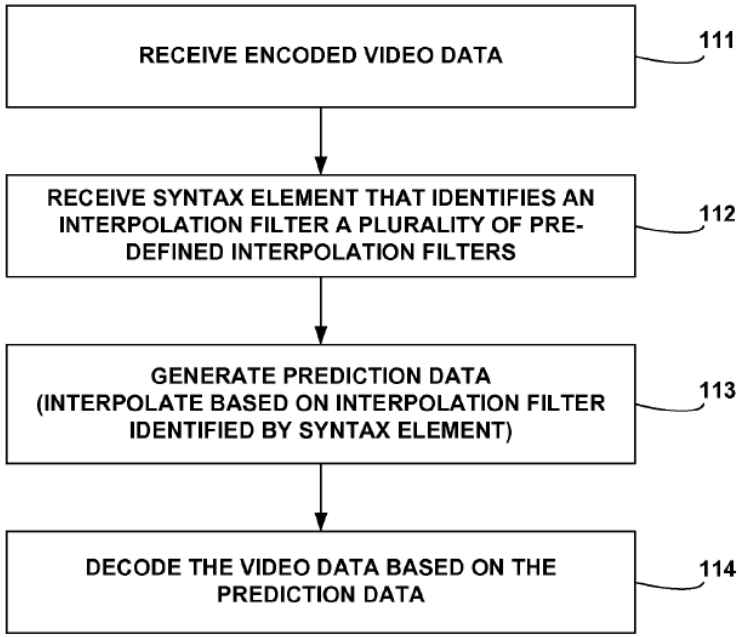
Nokia Exhibit 2006, p. 249
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 6

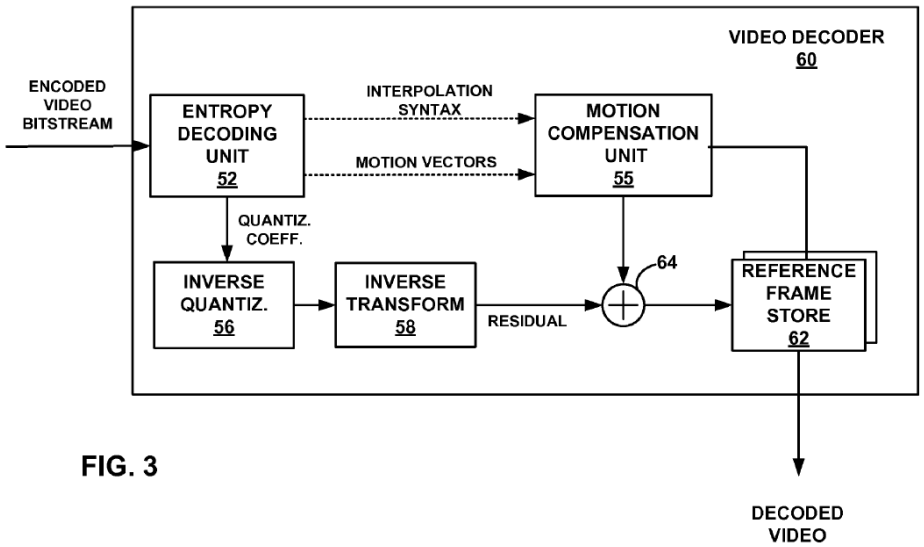
Nokia Exhibit 2006, p. 250
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz II, as evidenced by the example citations below, discloses reconstructing the block of pixels based on the combined prediction. *See, e.g.,* Karczewicz II, ¶0111: > 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. Karczewicz II, Fig. 6:

Nokia Exhibit 2006, p. 251
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 6 Karczewicz II, ¶0065:

Nokia Exhibit 2006, p. 252
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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). Karczewicz II, Fig. 3:  FIG. 3 Karczewicz II, ¶0006:

Nokia Exhibit 2006, p. 253
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 2; ; claim 19.*

Nokia Exhibit 2006, p. 254
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
using pixel values of said first reference block.	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 reference block.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 3.* *Supra claim 20.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 255
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim ; claim 20..* *Supra limitation 19[c].* Furthermore, left-shifting to multiply a value by a coefficient and increase the precision of intermediate values in calculations was well known in the art. *See, e.g.*, U.S. Patent App. Pub. 2008/0198935 ("Srinivasan") at ¶91, ¶116, ¶124, ¶129, ¶131; U.S. Patent App. Pub. No. 2013/0034158 ("Kirchhoffer-158") at ¶67, ¶¶84-85, ¶103. This confirms that it would have been obvious to shift values of the second reference block to the left. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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	The combination of Karczewicz I and Karczewicz II 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.

Nokia Exhibit 2006, p. 256
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
prediction to the right, further comprises: inserting a rounding offset to the combined prediction before said decreasing.	*See, e.g.,* *See claim 5.* *Supra* claim 19. *Supra* limitation 19[d]. *Infra* limitation 25[preamble]. *Infra* limitation 31[preamble]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 6.*

Nokia Exhibit 2006, p. 257
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
prediction and values of said second prediction.	*See limitation 19[a].* *Supra claim 19.* *Supra limitations 19[c]-19[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 I, as evidenced by the example citations below, discloses an apparatus for decoding a block of pixels. *See, e.g., See limitation 19[preamble].* Karczewicz I, ¶0003: > Digital multimedia capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless communication devices, 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, digital media players, and the like. Digital multimedia devices may implement video coding techniques, such as MPEG-2, ITU-H.263, MPEG-4, or ITU-H.264/MPEG-4 Part 10, Advanced Video Coding (AVC), to transmit and receive or store and retrieve digital video data more efficiently. Video encoding techniques may perform video

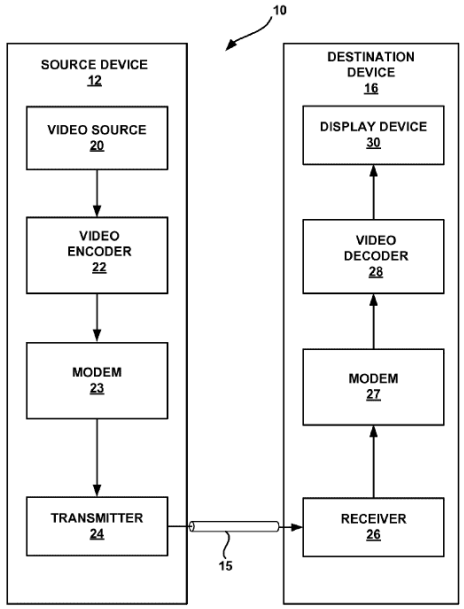
Nokia Exhibit 2006, p. 258
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	compression via spatial and temporal prediction to reduce or remove redundancy inherent in video sequences. Karczewicz I, ¶0012: 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. Karczewicz I, ¶¶0029-30: [0029] FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite radiotelephones, or any devices that can communicate video information over a communication channel 15, which may or may not be wireless. The techniques of this disclosure, however, which concern non-zero rounding and prediction mode selection techniques, are not necessarily limited to wireless applications or settings. [0030] 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 non-

Nokia Exhibit 2006, p. 259
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	zero rounding and prediction mode selection techniques as part of a video encoding process. Video decoder 28 may receive one or more syntax elements indicating the selection and indicating whether non-zero rounding was used. Accordingly, video decoder 28 may perform the proper weighted prediction signaled in the received bistream. Karczewicz I, ¶0033: Receiver 26 of destination device 16 receives information over channel 15, and modem 27 demodulates the information. Again, the video encoding process may implement one or more of the techniques described herein to provide non-zero rounding and prediction mode selection consistent with this disclosure. The information communicated over channel 15 may include information defined by video encoder 22, which may be used by video decoder 28 consistent with this disclosure. Display device 30 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a cathode ray tube, a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device. Karczewicz I, Fig. 1:

Nokia Exhibit 2006, p. 260
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 1 Karczewicz I, ¶¶0038-40: [0038] 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

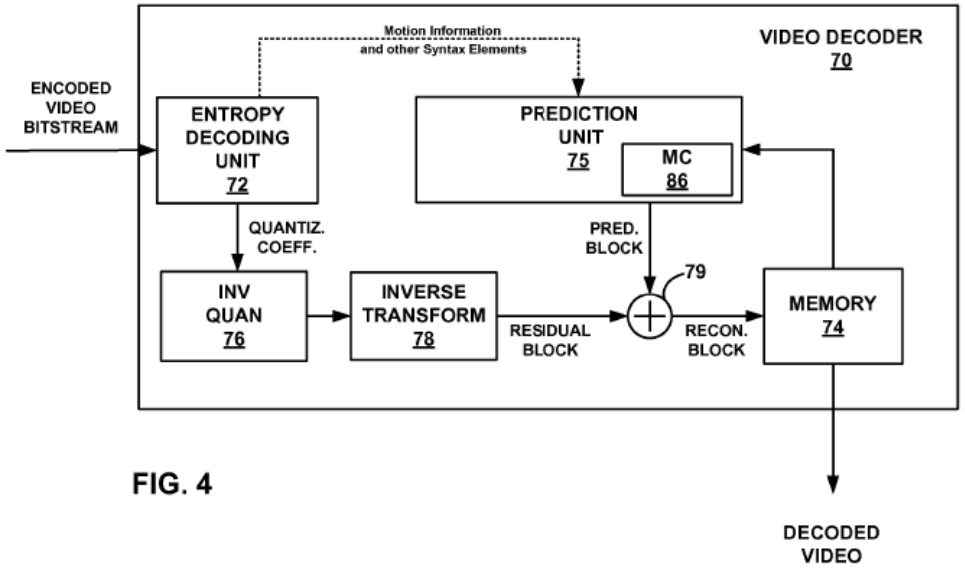
Nokia Exhibit 2006, p. 261
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	combined encoder/decoder (CODEC) in a respective mobile device, subscriber device, broadcast device, server, or the like. [0039] A video sequence typically includes a series of video frames. Video encoder 22 and video decoder 28 may operate on video blocks within individual video frames in order to encode and decode the video data. The video blocks may have fixed or varying sizes, and may differ in size according to a specified coding standard. Each video frame may include a series of slices or other independently decodable units. Each slice may include a series of macroblocks, which may be arranged into sub-blocks. As an example, the ITU-T 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. [0040] 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 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 macroblocks and/or sub-blocks. Each slice may be an independently decodable unit of a video frame. Alternatively, frames themselves may be decodable units, or other portions of a frame may be defined as decodable units. The term "coded unit" refers to any independently decodable unit of a video frame such as an entire frame, a slice of a frame, a group of pictures (GOPs), or another independently decodable unit defined according to the coding techniques used. Karczewicz I, ¶¶0083-86:

Nokia Exhibit 2006, p. 262
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0083] FIG. 4 is a block diagram illustrating an exemplary video decoder 70, which may perform the reciprocal decoding techniques to the encoding techniques described above. Video decoder 70 may include an entropy decoding unit 72, a prediction unit 75, an inverse quantization unit 76, an inverse transform unit 78, a memory 74 and adder 79. Prediction unit 75 may include a motion compensation (MC) unit 88, as well as spatial prediction components, which are not shown for simplicity and ease of illustration. [0084] Video decoder 70 may receive encoded video data, and one or more syntax elements that indicate whether a rounding adjustment was used to encode the encoded video data. MC unit 86 of prediction unit 75 may generate weighted prediction data that depends on two or more lists of data, as described herein. In accordance with this disclosure, the weighted prediction data does not include the rounding adjustment if the one or more syntax elements indicate that the rounding adjustment was not used to encode the encoded video data. Video decoder 70 can decode the video data using the weighted prediction data, e.g., by invoking adder 79 to add the weighted prediction data (e.g., a prediction block) to residual data (e.g., a residual block). [0085] In general, entropy decoding unit 72 receives an encoded bitsteam and entropy decodes the bitstream to generate quantized coefficients, motion information and other syntax. The motion information (e.g., motion vectors) and other syntax are forwarded to prediction unit 75 for use in generating the predictive data. Prediction unit 75 performs bidirectional prediction consistent with this disclosure, avoiding rounding adjustments in some cases, and possibly implementing default, implicit or explicit weighted prediction according to the received syntax elements. The syntax elements may identify the type of weighted prediction that to be used, coefficients and offset if explicit weighted prediction to be used, and whether rounding adjustments should be used in the decoding. [0086] The quantized coefficients are sent from entropy decoding unit 72 to inverse quantization unit 76, which performs inverse quantization. Inverse transform unit 78 then

Nokia Exhibit 2006, p. 263
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	inverse transforms the de-quantized coefficients back to the pixel domain to generate a residual block. Adder 79 combines the prediction data (e.g., a prediction block) generated by prediction unit 75 with the residual block from inverse transform unit 78 to create a reconstructed video block, which may be stored in memory 74 and/or output from video decoder 70 as decoded video output. Karczewicz I, Fig. 4:  FIG. 4

Nokia Exhibit 2006, p. 264
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Karczewicz I, ¶0098: The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units. The techniques described herein may also be implemented in hardware, software, firmware, or any combination thereof. Any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. In some cases, various features may be implemented as an integrated circuit device, such as an integrated circuit chip or chipset. Karczewicz I, ¶0099: If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. Karczewicz I, ¶0100:

Nokia Exhibit 2006, p. 265
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	The code or instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video codec. Also, the techniques could be fully implemented in one or more circuits or logic elements. Karczewicz II, as evidenced by the example citations below, discloses an apparatus for decoding a block of pixels. *See, e.g.,* *See limitation 19[preamble].* Karczewicz II, ¶0003: 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. Karczewicz II, ¶0015:

Nokia Exhibit 2006, p. 266
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 data based on a plurality of different pre-defined interpolation filters, and the video encoder encodes the video data based on the prediction data. Karczewicz II, ¶0016: 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. Karczewicz II, ¶0017: 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. Karczewicz II, ¶0022: 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

Nokia Exhibit 2006, p. 267
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶¶0023-24: [0023] 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 encoded by the motion compensation unit to another device. [0024] 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. Karczewicz II, ¶0025: 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

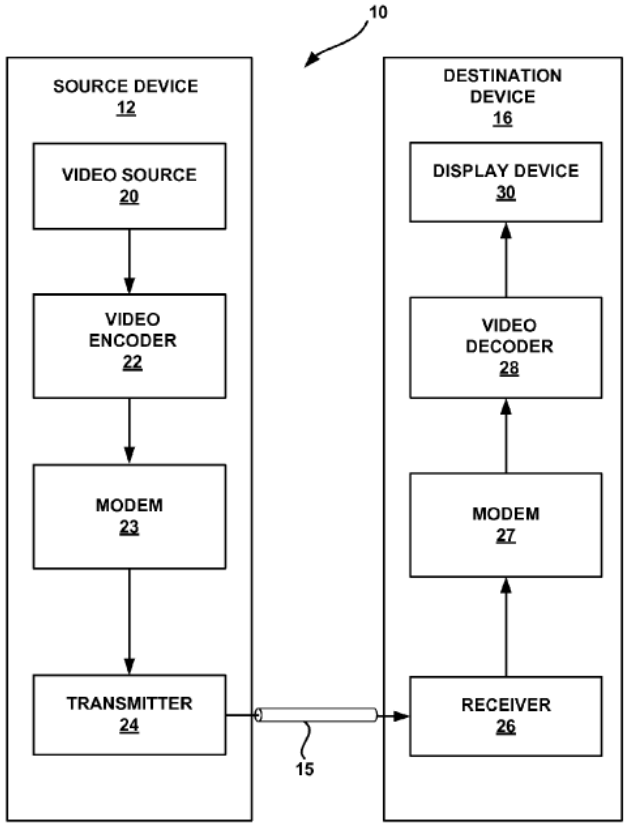
Nokia Exhibit 2006, p. 268
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶¶0026-27: [0026] 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. [0027] 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. Karczewicz II, ¶¶0040-41: [0041] FIG. 1 is a block diagram illustrating one exemplary video encoding and decoding system 10 that may implement techniques of this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that transmits encoded video to a destination device 16 via a communication channel 15. Source device 12 and destination device 16 may comprise any of a wide range of devices. In some cases, source device 12 and destination device 16 comprise wireless communication devices, such as wireless handsets, so-called cellular or satellite radiotelephones, or any wireless devices that can communicate video information over a

Nokia Exhibit 2006, p. 269
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	communication channel 15, in which case communication channel 15 is wireless. The techniques of this disclosure, however, which concern motion compensated interpolation, are not necessarily limited to wireless applications or settings. [0041] 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. Karczewicz II, ¶0044: Receiver 26 of destination device 16 receives information over channel 15, and modem 27 demodulates the information. Again, the video encoding process may implement one or more of the techniques described herein to improve the interpolation during motion compensation. The video decoding process performed by video decoder 28 may also perform interpolation during its motion compensation stage of the decoding process. Display device 30 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a cathode ray tube, a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device. Karczewicz II, Fig.1:

Nokia Exhibit 2006, p. 270
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	 FIG. 1 Karczewicz II, ¶0048:

Nokia Exhibit 2006, p. 271
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, Fig. 3:

Nokia Exhibit 2006, p. 272
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

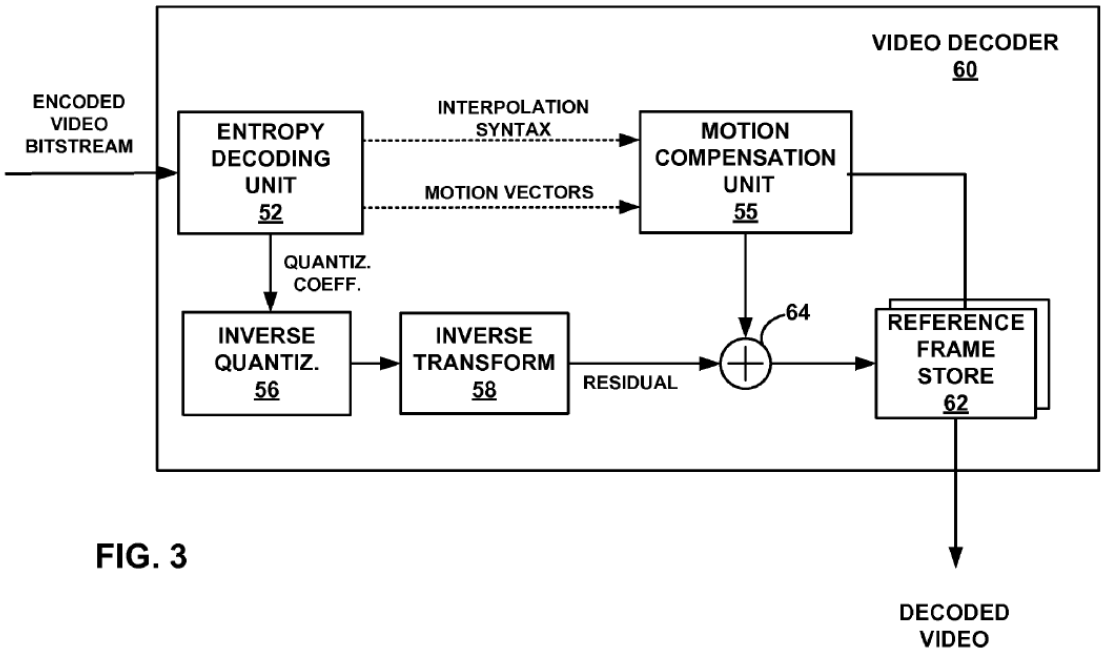


FIG. 3

Karczewicz II, ¶0050:

> 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

Nokia Exhibit 2006, p. 273
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶¶0063-66: [0063] 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 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. [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 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).

Nokia Exhibit 2006, p. 274
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	[0065] 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). [0066] 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 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. Karczewicz II, ¶¶0118-120: [0118] The techniques of this disclosure may be embodied in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units, etc. [0119] Accordingly, the techniques described herein may be implemented in hardware, software, firmware, or any combination thereof. If implemented in hardware, any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described

Nokia Exhibit 2006, p. 275
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. [0120] The code may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video encoder-decoder (CODEC). Also, the techniques could be fully implemented in one or more circuits or logic elements. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 276
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
25[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 I, 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. *See, e.g.,* *See limitation 7[a].* Karczewicz I, ¶0005: > For inter-coding, a video encoder performs motion estimation to track the movement of matching video blocks between two or more adjacent frames or other coded units, such as slices of frames. Inter-coded frames may include predictive frames ("P-frames"), which may include blocks predicted from a previous frame, and bidirectional predictive frames ("B-frames"), which may include blocks predicted from a previous frame and a subsequent frame of a video sequence. The terms P-frames and B-frames are somewhat historic in the sense that early coding techniques limited prediction in specific directions. Newer coding formats and standards may not limit the prediction direction of P-frames or B-frames. Thus, the term "bi-directional" now refers to prediction based on two or more lists of reference data regardless of the temporal relationship of such reference data relative to the data being coded. Karczewicz I, ¶0006: > Consistent with newer video standards such as ITU H.264, for example, bi-directional prediction may be based on two different lists which do not necessarily need to have data that resides temporally before and after the current video block. In other words, B-video blocks may be predicted from two lists of data, which may correspond to data from two previous frames, two subsequent frames, or one previous frame and one subsequent frame. In contrast, P-video blocks are predicted based on one list, i.e., one data structure, which may correspond to one

Nokia Exhibit 2006, p. 277
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	predictive frame, e.g., one previous frame or one subsequent frame. B-frames and P-frames may be more generally referred to as P-units and B-units. P-units and B-units may also be realized in smaller coded units, such as slices of frames or portions of frames. B-units may include B-video blocks, P-video blocks or I-video blocks. P-units may include P-video blocks or I-video blocks. I-units may include only I-video blocks. Karczewicz I, ¶0022: This disclosure describes video encoding and decoding techniques applicable to bi-directional prediction. In bi-directional prediction, a video block is predictively encoded and decoded based on two different lists of predictive reference data. In one aspect of this disclosure, rounding adjustments to bi-directional predictive data may be purposely eliminated to provide predictive data that lacks any rounding bias. In this case, rounded and unrounded predictive data may both be considered in a rate-distortion analysis to identify the best data for prediction of a given video block. One or more syntax elements may be encoded to indicate the selection, and a decoder may interpret the one or more syntax elements in order to determine whether rounding should be used in the decoding process. Karczewicz I, ¶0042: The techniques of this disclosure are specifically applicable to weighted bi-directional prediction. As mentioned above, bi-directional prediction is prediction of so-called "B-video blocks" based on two different lists of data. B-video blocks may be predicted from two lists of data from two previous frames, two lists of data from subsequent frames, or one list of data from a previous frame and one from a subsequent frame. In contrast, P-video blocks are predicted based on one list, which may correspond to one predictive frame, e.g., one previous frame or one subsequent frame. B-frames and P-frames may be more generally referred to as P-units and B-units. P-units and B-units may also be realized in smaller coded units, such as slices of frames or portions of frames. B-units may include B-video blocks, P-video blocks or

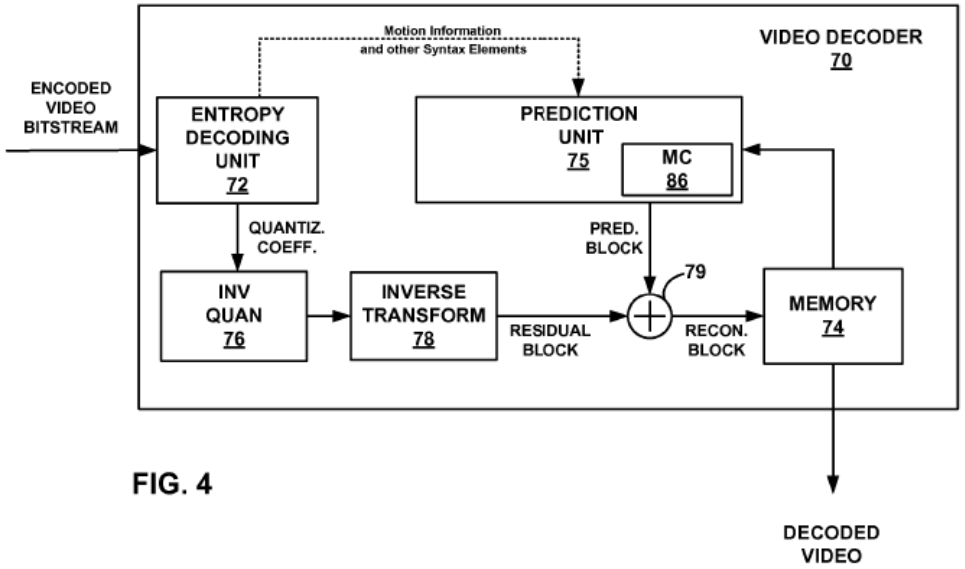
Nokia Exhibit 2006, p. 278
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I-video blocks. P-units may include P-video blocks or I-video blocks. I-units may include only I-video blocks. Karczewicz I, ¶0007: For P- and B-video blocks, motion estimation generates motion vectors, which indicate the displacement of the video blocks relative to corresponding prediction video blocks in predictive reference frame(s) or other coded units. Motion compensation uses the motion vectors to generate prediction video blocks from the predictive reference frame(s) or other coded units. After motion compensation, a residual video block is formed by subtracting the prediction video block from the original video block to be coded. The video encoder usually applies transform, quantization and entropy coding processes to further reduce the bit rate associated with communication of the residual block. I-and P-units are commonly used to define reference blocks for the inter-coding of P- and B-units. Karczewicz I, ¶0012: 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. Karczewicz I, ¶0038: 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,

Nokia Exhibit 2006, p. 279
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 encoder/decoder (CODEC) in a respective mobile device, subscriber device, broadcast device, server, or the like. Karczewicz I, ¶0085: In general, entropy decoding unit 72 receives an encoded bitsteam and entropy decodes the bitstream to generate quantized coefficients, motion information and other syntax. The motion information (e.g., motion vectors) and other syntax are forwarded to prediction unit 75 for use in generating the predictive data. Prediction unit 75 performs bidirectional prediction consistent with this disclosure, avoiding rounding adjustments in some cases, and possibly implementing default, implicit or explicit weighted prediction according to the received syntax elements. The syntax elements may identify the type of weighted prediction that to be used, coefficients and offset if explicit weighted prediction to be used, and whether rounding adjustments should be used in the decoding. Karczewicz I, ¶0088: FIG. 6 is a flow chart illustrating an exemplary process performed by a video decoder consistent with this disclosure. FIG. 6 will be described from the perspective of video decoder 70 of FIG. 4. As shown in FIG. 6, video decoder receives encoded video data (111), and receives one or more syntax elements that indicate whether a rounding adjustment was used to encode the video data (1 12). In particular, entropy decoding unit 72 may receive an encoded bitstream that includes the video data and the one or more syntax elements. Following entropy decoding, entropy decoding unit 72 may output the video data as quantized transform coefficients, which are inverse quantized by unit 76 and inverse transformed by unit 78. Entropy decoding unit 72 may output syntax elements to prediction unit, which includes the one or more syntax elements

Nokia Exhibit 2006, p. 280
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	that indicate whether a rounding adjustment was used to encode the video data, motion vectors and possibly other syntax. Karczewicz I, Fig. 4:  FIG. 4 Karczewicz I, ¶0053: 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

Nokia Exhibit 2006, p. 281
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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 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. For demonstrative purposes, the techniques described in this disclosure are described as being performed by motion compensation unit 35. Karczewicz I, ¶0054: Motion estimation unit 32 selects the appropriate motion vector for the video block to be coded by comparing the video block to video blocks of one or more predictive coded units (e.g., a previous and/or future frame in terms of time or temporally). Motion estimation unit 32 may, as an example, select a motion vector for a B-frame in a number of ways. In one way, motion estimation unit 32 may select a previous or future frame from a first set of frames (referred to as list 0) and determine a motion vector using only this previous or future frame from list 0. Alternatively, motion estimation unit 32 may select a previous or future frame from a second set of frames (referred to as list 1) and determine a motion vector using only this previous or future frame from list 1. In yet another way, motion estimation unit 32 may select a first frame from list 0 and a second frame from list 1 and select one or more motion vectors from the first frame of list 0 and the second frame of list 1. This third form of prediction may be referred to as bi-predictive motion estimation. Techniques of this disclosure may be implemented so as to efficiently select a motion-compensated bi-prediction mode. The selected motion vector for any given list may point to a predictive video block that is most similar to the video block being coded, e.g., as defined by a metric such as sum of absolute difference (SAD) or sum of squared

Nokia Exhibit 2006, p. 282
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	difference (SSD) of pixel values of the predictive block relative to pixel values of the block being coded. Karczewicz I, ¶0098: The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, and integrated circuit (IC) or a set of ICs (i.e., a chip set). Any components, modules or units have been described provided to emphasize functional aspects and does not necessarily require realization by different hardware units. The techniques described herein may also be implemented in hardware, software, firmware, or any combination thereof. Any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. In some cases, various features may be implemented as an integrated circuit device, such as an integrated circuit chip or chipset. Karczewicz I, ¶0099: If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or

Nokia Exhibit 2006, p. 283
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. Karczewicz I, ¶0100: The code or instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video codec. Also, the techniques could be fully implemented in one or more circuits or logic elements. 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. *See, e.g.,* Karczewicz II, ¶0006: 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

Nokia Exhibit 2006, p. 284
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0025: 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. Karczewicz II, ¶0026: 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. Karczewicz II, ¶0048: 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

Nokia Exhibit 2006, p. 285
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0063: 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 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. Karczewicz II, Fig. 3:

Nokia Exhibit 2006, p. 286
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]

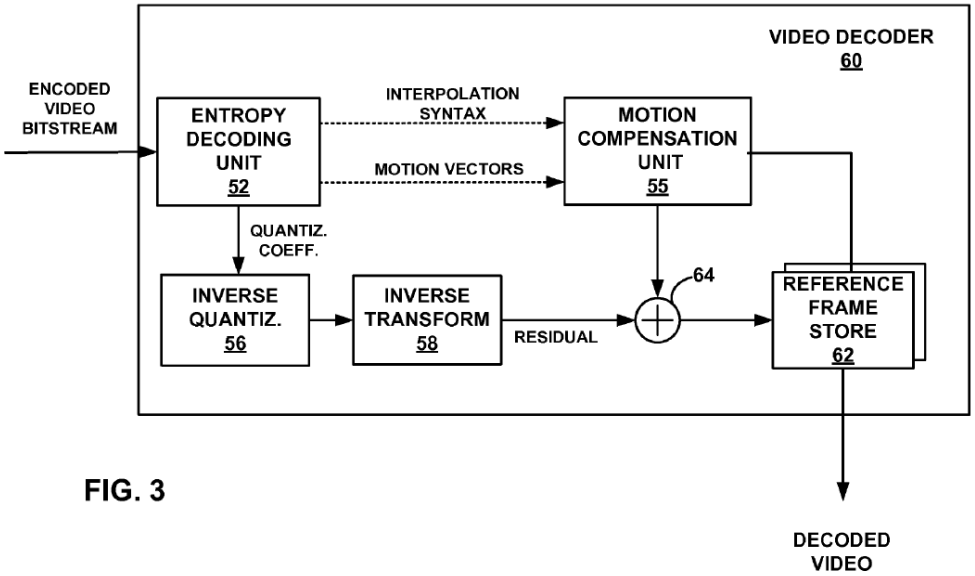


FIG. 3

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 produces motion compensated blocks in the manner described herein, e.g., including interpolation based

Nokia Exhibit 2006, p. 287
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	on a set of interpolation filter coefficients identified by the syntax element (i.e., the interpolation syntax). Karczewicz II, ¶0110: 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. Karczewicz II, ¶¶0119-120: [0119] Accordingly, the techniques described herein may be implemented in hardware, software, firmware, or any combination thereof. If implemented in hardware, any features described as modules, units or components may be implemented together in an integrated logic device or separately as discrete but interoperable logic devices. If implemented in software, the techniques may be realized at least in part by a computer-readable medium comprising instructions that, when executed in a processor, performs one or more of the methods described above. The computer-readable medium may comprise a computer-readable storage medium and may form part of a computer program product, which may include packaging materials. The computer-readable storage medium may comprise random access memory (RAM) such as synchronous dynamic random access memory (SDRAM), read-only memory (ROM), non-volatile random access memory (NVRAM), electrically erasable programmable read-only

Nokia Exhibit 2006, p. 288
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	memory (EEPROM), FLASH memory, magnetic or optical data storage media, and the like. The techniques additionally, or alternatively, may be realized at least in part by a computer-readable communication medium that carries or communicates code in the form of instructions or data structures and that can be accessed, read, and/or executed by a computer. [0120] The code may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, an application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated software modules or hardware modules configured for encoding and decoding, or incorporated in a combined video encoder-decoder (CODEC). Also, the techniques could be fully implemented in one or more circuits or logic elements. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[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	The combination of Karczewicz I and Karczewicz II 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.

Nokia Exhibit 2006, p. 289
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;	*See, e.g.*, *See limitation 19[a].* *Supra limitation 25[a].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[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;	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.*, *See limitation 19[b].* *Supra limitation 25[b].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

Nokia Exhibit 2006, p. 290
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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[d] use said second reference block to obtain a second prediction, said second prediction having the second precision;	The combination of Karczewicz I and Karczewicz II discloses using the second reference block to obtain a second prediction, the second prediction having the second precision. *See, e.g.,* *See limitation 19[c].* *Supra limitations 25[b]-25[c].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[e] obtain a combined prediction based at least partly upon said first prediction and said second prediction;	The combination of Karczewicz I and Karczewicz II discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction. *See, e.g.,* *See limitation 19[d].*

Nokia Exhibit 2006, p. 291
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Supra limitations 25[b]-25[d]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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[f] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and	The combination of Karczewicz I and Karczewicz II discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right. *See, e.g.,* *See limitation 19[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 292
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
25[g] reconstruct the block of pixels based on the combined prediction.	The combination of Karczewicz I and Karczewicz II discloses reconstructing the block of pixels based on the combined prediction. *See* limitation 19[f]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
26. The apparatus according to claim 25, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 20.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 293
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 21*. *Supra claim 26.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 22.* *Supra claim 26.*

Nokia Exhibit 2006, p. 294
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Supra limitation 19[c]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
29. The apparatus according to claim 25, wherein the at least one memory and computer code are configured to cause the apparatus to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, by: inserting a rounding offset to the combined prediction before said decreasing.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 23.* *Supra claim 25.* *Supra limitation 19[d].* *Supra limitation 25[preamble].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*.

Nokia Exhibit 2006, p. 295
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.	The combination of Karczewicz I and Karczewicz II disclosesthat 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. *See, e.g.,* *See claim 24.* *See limitation 25[b].* *Supra claim 25.* *Supra limitations 19[c]-19[e].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 296
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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 comprising program code instructions configured to:	Karczewicz I, 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. *See, e.g.,* *See limitations 19[preamble], 25[preamble], 25[a].* Karczewicz I, ¶0013: > Accordingly, this disclosure also contemplates a computer-readable storage medium comprising instructions that when executed cause a processor to select between default weighted prediction and implicit weighted prediction, upon selecting default weighted prediction, select between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the default weighted prediction, upon selecting implicit weighted prediction, select between explicit weighted prediction based on calculated weights and explicit weighted prediction based on weights defined by the implicit weighted prediction, and encode the video data using weighed prediction data defined by the selections. 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. *See, e.g.,* Karczewicz II, ¶0028: > 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

Nokia Exhibit 2006, p. 297
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	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. Karczewicz II, ¶0027: 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. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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	The combination of Karczewicz I and Karczewicz II 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.

Nokia Exhibit 2006, p. 298
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
vector, wherein the pixels of the current block, the first reference block, and the second reference block have values with a first precision;	*See, e.g.*, *See limitation 19[a].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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;	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.*, *See limitation 19[b].* *Supra limitation 31[a].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 299
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
31[c] use said second reference block to obtain a second prediction, said second prediction having the second precision;	The combination of Karczewicz I and Karczewicz II discloses using the second reference block to obtain a second prediction, the second prediction having the second precision. *See, e.g.,* *See limitation 19[c].* *Supra limitation 31[b].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
31[d] obtain a combined prediction based at least partly upon said first prediction and said second prediction;	The combination of Karczewicz I and Karczewicz II discloses obtaining a combined prediction based at least partly upon the first prediction and the second prediction. *See, e.g.,* *See limitation 19[d].* *Supra limitations 31[b]-31[c].*See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

Nokia Exhibit 2006, p. 300
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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.
31[e] decrease a precision of said combined prediction by shifting bits of the combined prediction to the right; and	The combination of Karczewicz I and Karczewicz II discloses decreasing a precision of the combined prediction by shifting bits of the combined prediction to the right. *See, e.g.,* *See limitation 19[e].* *Supra limitations 31[b]-31[d].* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
31[f] reconstruct the block of pixels based on the combined prediction.	The combination of Karczewicz I and Karczewicz II discloses reconstructing the block of pixels based on the combined prediction. *See, e.g.,* *See limitation 19[f].*

Nokia Exhibit 2006, p. 301
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Supra limitations 19[b]-19[d]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
32. The computer program product according to claim 31, wherein in an instance in which said first motion vector points to a subpixel, said first prediction is obtained by interpolation using pixel values of said first reference block.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 20*. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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 2006, p. 302
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 21.* *Supra claim 31.* See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 22.* *Supra claim 32.*

Nokia Exhibit 2006, p. 303
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	Supra limitation 19[c]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
35. The computer program product according to claim 31, wherein the program code instructions configured to decrease said precision of said combined prediction by shifting bits of the combined prediction to the right, further comprise program code instructions configured to: insert a rounding offset to the combined prediction before said decreasing.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claims 23 and 29.* *Supra* claim 31. *Supra* limitation 19[d]. *Supra* limitation 25[preamble]. *Supra* limitation 31[preamble].

Nokia Exhibit 2006, p. 304
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz I and 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.*
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.	The combination of Karczewicz I and Karczewicz II 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. *See, e.g.,* *See claim 24.* *See limitation 31[a].* *Supra* claim 31. *Supra* limitations 19[c]-19[e]. See also the disclosures and analysis of Karczewicz I and Karczewicz II from "The Combination of Karczewicz I and Karczewicz II," set forth *supra*. *Further, to the extent Complainant contends this element is not expressly or inherently disclosed by Karczewicz I and Karczewicz II, it would have been obvious in view of the disclosures in Karczewicz*

Nokia Exhibit 2006, p. 305
ASUS v. Nokia
IPR2025-01154

ASSERTED CLAIMS	PRIOR ART REFERENCE: "KARCZEWICZ I" + "KARCZEWICZ II"[2]
	I and 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 2006, p. 306
ASUS v. Nokia
IPR2025-01154