METHOD AND APPARATUS FOR VIDEO CODING

[0001] This present application claims the benefit of U.S. Patent Application No. 16/822,075, "Method and Apparatus for Video coding" filed on March 18, 2020, which claims the benefit of U.S. Provisional Application No. 62/820,196, "Affine Inter Prediction Refinement Methods" filed on March 18, 2019, No. 62/828,425, "LIC Signaling and Affine Refinement" filed on April 2, 2019, No. 62/838,798, "Inter Prediction Refinement Methods" filed on April 25, 2019.

TECHNICAL FIELD

[0002] The present disclosure describes embodiments generally related to video coding.

BACKGROUND

[0003] The background description provided herein is for the purpose of generally presenting the context of the disclosure. Work of the presently named inventors, to the extent the work is described in this background section, as well as aspects of the description that may not otherwise qualify as prior art at the time of filing, are neither expressly nor impliedly admitted as prior art against the present disclosure.

[0004] Video coding and decoding can be performed using inter-picture prediction with motion compensation. Uncompressed digital video can include a series of pictures, each picture having a spatial dimension of, for example, 1920 x 1080 luminance samples and associated chrominance samples. The series of pictures can have a fixed or variable picture rate (informally also known as frame rate), of, for example 60 pictures per second or 60 Hz. Uncompressed video has significant bitrate requirements. For example, 1080p60 4:2:0 video at 8 bit per sample (1920x1080 luminance sample resolution at 60 Hz frame rate) requires close to 1.5 Gbit/s bandwidth. An hour of such video requires more than 600 GBytes of storage space.

[0005] One purpose of video coding and decoding can be the reduction of redundancy in the input video signal, through compression. Compression can help reduce the aforementioned bandwidth or storage space requirements, in some cases by two orders of magnitude or more. Both lossless and lossy compression, as well as a combination thereof can be employed. Lossless compression refers to techniques where an exact copy of the original signal can be reconstructed from the compressed original signal. When using lossy

prediction mechanisms that H.265 offers, described here is a technique henceforth referred to as "spatial merge".

[0009] Referring to FIG. 1, a current block (101) comprises samples that have been found by the encoder during the motion search process to be predictable from a previous block of the same size that has been spatially shifted. Instead of coding that MV directly, the MV can be derived from metadata associated with one or more reference pictures, for example from the most recent (in decoding order) reference picture, using the MV associated with either one of five surrounding samples, denoted A0, A1, and B0, B1, B2 (102 through 106, respectively). In H.265, the MV prediction can use predictors from the same reference picture that the neighboring block is using.

SUMMARY

[0010] Aspects of the disclosure provide methods and apparatuses for video encoding/decoding. In some examples, an apparatus for video decoding includes processing circuitry. The circuitry can be configured to determine whether to apply the PROF to an affine coded block, and responsive to a determination to apply the PROF to the affine coded block, perform a PROF process.

[0011] For example, during the PROF process, a prediction sample I(i,j) at a sample location (i,j) in the affine coded block can be generated. Spatial gradients g_x (i,j) and g_y (i,j) at the sample location (i,j) in the affine coded block can be generated. A prediction refinement ΔI(i,j) based on the spatial gradients g_x (i,j) and g_y (i,j) can be generated. The prediction refinement ΔI(i,j) can added to the prediction sample I(i,j) to generate a refined prediction sample.

[0012] In an example, a syntax element indicating whether the PROF is enabled for affine prediction is received at the decoder. In an example, the PROF is enabled for affine prediction by default.

[0013] In an example, the syntax element is signaled at a sequence level, a slice level, a tile level, a tile group level, or a picture level. In an example, whether to apply the PROF to the affine coded block can be determined based on values of affine parameters of an affine model of the affine coded block. In an example, the spatial gradients g_x (i,j) and g_y (i,j) at the sample location (i,j) are generated based on a first prediction sample(s) of a first sub-block including the prediction sample I(i,j) and a second prediction sample(s) of a second sub-block neighboring the first sub-block. The first sub-block and the second sub-block can be partitioned from the affine coded block. In an example, inter prediction for sub-blocks of

Direction IDX	00	01	10	11
x-axis	+	–	N/A	N/A
y-axis	N/A	N/A	+	–

[0135] It is noted that the meaning of an MVD sign can vary according to the information of starting MV(s). When the starting MV(s) is a uni-prediction MV or bi-prediction MVs both pointing to the same side of a current picture (i.e. POCs of two references are both larger than the POC of the current picture, or are both smaller than the POC of the current picture), the sign in Table 2 specifies the sign of MV offset added to the starting MV. When the starting MVs are bi-prediction MVs with the two MVs pointing to different sides of the current picture (i.e. the POC of one reference is larger than the POC of the current picture, and the POC of the other reference is smaller than the POC of the current picture), the sign in Table 2 specifies the sign of MV offset added to the L0 MV component of the starting MV and the sign for the L1 MV has an opposite value.

[0136] Based on the basis MV, the offset, and the MVD sign, the final MV(s) can be determined for the current CU.

[0137] **3. Affine motion compensated prediction**

[0138] In some examples, a translational motion model is applied for motion compensation prediction (MCP). However, the translational motion model may not be suitable for modeling other types of motions, such as zoom in/out, rotation, perspective motions, and the other irregular motions. In some embodiments, a block-based affine transform motion compensation prediction is applied. In FIG. 13A, an affine motion field of a block is described by two control point motion vectors (CPMVs), CPMV0 and CPMV1, of two control points (CPs), CP0 and CP1 when a 4-parameter affine model is used. In FIG. 13B, an affine motion field of a block is described by three CPMVs, CPMV0, CPMV1 and CPMV3, of CPs, CP0, CP1, and CP2 when a 6-parameter affine model is used.

[0139] For a 4-parameter affine motion model, a motion vector at a sample location (x, y) in a block is derived as:

$$\begin{cases} mv_x = \frac{mv_{1x}-mv_{0x}}{W}x + \frac{mv_{1y}-mv_{0y}}{W}y + mv_{0x} \\ mv_y = \frac{mv_{1y}-mv_{0y}}{W}x + \frac{mv_{1y}-mv_{0x}}{W}y + mv_{0y} \end{cases} \quad \text{(Eq. 1)}$$

[0140] For a 6-parameter affine motion model, a motion vector at sample location (x, y) in a block is derived as:

$$\begin{cases} mv_x = \frac{mv_{1x}-mv_{0x}}{W}x + \frac{mv_{2x}-mv_{0x}}{H}y + mv_{0x} \\ mv_y = \frac{mv_{1y}-mv_{0y}}{W}x + \frac{mv_{2y}-mv_{0y}}{H}y + mv_{0y} \end{cases} \quad \text{(Eq. 2)}$$

[0141] In the expressions (1) and (2), (mv_{0x}, mv_{0y}) is a motion vector of the top-left corner control point, (mv_{1x}, mv_{1y}) is motion vector of the top-right corner control point, and (mv_{2x}, mv_{2y}) is motion vector of the bottom-left corner control point. In addition, the coordinate (x, y) is with respect to the top-left corner of the respective block, and W and H denotes the width and height of the respective block.

[0142] In order to simplify the motion compensation prediction, a sub-block based affine transform prediction is applied in some embodiments. For example, in FIG. 14, the 4-parameter affine motion model is used, and two CPMVs, $\overrightarrow{v_0}$ and $\overrightarrow{v_1}$, are determined. To derive a motion vector of each 4×4 (samples) luma sub-block (1402) partitioned from the current block (1410), a motion vector (1401) of the center sample of each sub-block (1402) is calculated according to above expressions (1), and rounded to a 1/16 fraction accuracy. Then, motion compensation interpolation filters are applied to generate a prediction of each sub-block (1402) with the derived motion vector (1401). The sub-block size of chroma-components is set to be 4×4. A MV of a 4×4 chroma sub-block is calculated as the average of the MVs of the four corresponding 4×4 luma sub-blocks.

[0143] Similar to translational motion inter prediction, two affine motion inter prediction modes, affine merge mode and affine AMVP mode, are employed in some embodiments.

[0144] 3.1 **Affine merge prediction**

[0145] In some embodiments, an affine merge mode can be applied for CUs with both width and height larger than or equal to 8. Affine merge candidates of a current CU is generated based on motion information of spatial neighboring CUs. There can be up to five affine merge candidates and an index is signaled to indicate the one to be used for the current CU. For example, the following three types of affine merge candidates are used to form an affine merge candidate list:

(i) Inherited affine merge candidates that are extrapolated from CPMVs of the neighbor CUs;

(ii) Constructed affine merge candidates that are derived using the translational MVs of the neighbor CUs; and

(iii) Zero MVs.

[0146] In some embodiments, there can be at most two inherited affine candidates which are derived from affine motion models of the neighboring blocks, one from left neighboring CUs and one from above neighboring CUs. The candidate blocks, for example, can be located at positions shown in FIG. 8. For the left predictor, the scan order is A0->A1,

17A shows a current block (1711) and its spatial neighboring blocks denoted A0, A1, B0, and B1 (1712, 1713, 1714, and 1715, respectively). In some examples, spatial neighboring blocks A0, A1, B0, and B1 and the current block (1711) belong to a same picture.

[0159] FIG. 17B is a schematic illustration of determining motion information for sub-blocks of the current block (1711) using the SbTMVP method based on a selected spatial neighboring block, such as block A1 in this non-limiting example, in accordance with an embodiment. In this example, the current block (1711) is in a current picture (1710), and a reference block (1761) is in a reference picture (1760) and can be identified based on a motion shift (or displacement) between the current block (1711) and the reference block (1761) indicated by a motion vector (1722).

[0160] In some embodiments, similar to a temporal motion vector prediction (TMVP) in HEVC, a SbTMVP uses the motion field in the collocated picture to improve motion vector prediction and merge mode for CUs in the current picture. In some embodiments, the same reference picture used by TMVP can be used for SbTMVP. In some embodiments, TMVP predicts motion information at a CU level but SbTMVP predicts motion at a sub-CU level. In some embodiments, TMVP uses the temporal motion vectors from collocated block in the reference picture, which has a corresponding position adjacent to a lower-right corner or a center of a current block, and SbTMVP uses the temporal motion vectors from a reference block, which can be identified by performing a motion shift based on a motion vector from one of the spatial neighboring blocks of the current block.

[0161] For example, as shown in FIG. 17A, neighboring blocks A1, B1, B0, and A0 can be sequentially checked in a SbTMVP process. As soon as a first spatial neighboring block that has a motion vector that uses the reference picture (1760) as its reference picture is identified, such as block A1 having the motion vector (1722) that points to a reference block AR1 in the reference picture (1760) for example, this motion vector (1722) can be used for performing the motion shift. If no such motion vector is available from the spatial neighboring blocks A1, B1, B0, and A0, the motion shift is set to (0, 0).

[0162] After determining the motion shift, the reference block (1761) can be identified based on a position of the current block (1711) and the determined motion shift. In FIG. 17B, the reference block (1761) can be further divided into 16 sub-blocks with reference motion information MRa through MRp. In some examples, the reference motion information for each sub-block in the reference block (1761) can be determined based on a smallest motion grid that covers a center sample of such sub-block. The motion information can include motion vectors and corresponding reference indices. The current block (1711) can be

[0172] Afterwards, a prediction refinement can be calculated by the following optical flow equation:

$$\Delta I(i,j) = g_x(i,j) * \Delta v_x(i,j) + g_y(i,j) * \Delta v_y(i,j), \quad \text{(Eq. 5)}$$

where the $\Delta v(i,j)$ (e.g., ΔV (1842)) is the difference between a pixel MV (e.g. pixel MV (1840)) for a sample location (i,j), denoted by $v(i,j)$, and the sub-block MV of the sub-block (e.g., sub-block MV 1820) to which the pixel location (i,j) belongs. Since the affine model parameters and the pixel location relative to the sub-block center are not changed from sub-block to sub- block, $\Delta v(i,j)$ can be calculated for the first sub-block, and reused for other sub-blocks in the same CU. In some examples, let x and y be the horizontal and vertical position of $\Delta v(i,j)$ with respect to the center of the sub-block, $\Delta v(i,j)$ can be derived by the following equation,

$$\begin{cases} \Delta v_x(x,y) = a * x + b * y \\ \Delta v_y(x,y) = c * x + d * y \end{cases}$$

(Eq. 6)

where $\Delta v_x(x,y)$ is the x component of $\Delta v(i,j)$, and $\Delta v_y(x,y)$ is the y component of $\Delta v(i,j)$.

[0173] For a 4-parameter affine model,

$$\begin{cases} a = d = \frac{v_{1x} - v_{0x}}{w} \\ c = -b = \frac{v_{1y} - v_{0y}}{w} \end{cases}, \text{ and}$$

(Eq. 7)

for a 6-parameter affine model,

$$\begin{cases} a = \frac{v_{1x} - v_{0x}}{w} \\ b = \frac{v_{2x} - v_{0x}}{h} \\ c = \frac{v_{1y} - v_{0y}}{w} \\ d = \frac{v_{2y} - v_{0y}}{h} \end{cases}$$

(Eq. 8)

where (v_{0x}, v_{0y}), (v_{1x}, v_{1y}), and (v_{2x}, v_{2y}) are the upper-left, upper-right, and lower-left control point motion vectors, and w and h are the width and height of a current block including the current sub-block.

[0174] Finally, the prediction refinement $\Delta I(i,j)$ can be added to the initial sub-block prediction sample I(i, j). The final refined prediction sample I' according to the PROF method can be generated using the following equation:

$$I'(i,j) = I(i,j) + \Delta I(i,j). \quad \text{(Eq. 9)}$$

range may be denoted as [M, N], where M and N are positive numbers. In one example, M and N are predefined, or signaled in bitstream, for example, in a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, a slice header, or the like.

[0214] In another embodiment, for an affine coded block, the usage (enable or disable) of the PROF may be determined by a flag, for example, denoted by prof_flag. The flag can be signaled at a bock level. Accordingly, the PROF can be enabled when the value of the flag is true.

[0215] In another embodiment, a syntax element may be used to determine whether the PROF for affine prediction is enabled or not. The syntax element can be a high level syntax element in an example. For example, the syntax element may be signaled at a sequence level, a slice level, a tile level, a tile group level, a picture level, or the like.

[0216] **2. Gradient calculation for PROF**

[0217] In some embodiments, in a PROF process, gradient calculation can be performed in a sub-block basis. For example, inter prediction (including interpolation operations during the inter prediction) can be performed for each affine sub-block (for example, for each 4x4 sub-block for affine prediction). The gradient is then calculated with a 3-tap filter [1 0 -1] based on the prediction output of the affine sub-block. Refinement is also applied on the sub-block subsequently. To calculate the gradient for each affine sub-block, the affine sub-block can be padded to a larger size, for example, extending 1 pixel on each side.

[0218] **2.1 Gradient calculation for an entire block**

[0219] In one embodiment, the gradient calculation in PROF may be performed for the entire coding block when the inter prediction (interpolation) for all affine sub-blocks are completed. In such a way, extending each sub-block for the gradient calculation can be avoided. Instead, the whole coding block can be extended (padded) for 1 pixel, for example, on each side for the gradient calculation, which is less complex than extending each sub-block. The refinement may be done for each sub-block based on the gradient of the respective sub-block obtained from the whole-block based calculation.

[0220] For the padding for the whole block, the following methods may be used in various embodiments: (1) The pixels on the extended borders are generated by interpolation using the corresponding sub-block's motion vector. (2)The pixels on the extended borders are copied from the nearest integer pixel position in the reference picture. (3) The pixels on

[0244] In an example, the PROF alike refinement process can include the following steps:

Step1: Forming a prediction block *Pred* with MV_{round} which represents a MV rounded to the highest precision supported by an existing interpolation filter.

Step2: Calculating a PROF alike prediction refinement offset $\Delta I(i,j)$ per sample location for the current block. One example of $\Delta I(i,j)$ is derived with the following formula:

$$\Delta I(i,j) = g_x(i,j) * \Delta v_x + g_y(i,j) * \Delta v_y \quad \text{(Eq. 22)}$$

where $g_x(i,j)$ and $g_y(i,j)$ are gradients of the prediction *Pred* at location (i, j) along x and y directions, respectively, and Δv_x and Δv_y are MV differences between MV_{block} and MV_{round}.

Step3: Adding $\Delta I(i,j)$ to inter prediction *Pred* and, in some examples, clipping the resulting pixels to target range if necessary.

[0245] In an embodiment, when the current block has the same MV for the whole block (e.g., sub-blocks having the same MV), the above PROF alike refinement process is applied to the whole block. When the current block has different motion vectors at sub-block level, such as blocks coded with SbTMVP, the above PROF alike refinement process is applied for each individual sub-block.

[0246] In an embodiment, the precision of MV_{block} may be predefined or signaled in a bitstream, such as in a sequence parameter set (e.g., SPS), a picture parameter set (e.g., PPS), a slice header, a tile group header, or the like. Alternatively, when BDOF is used, the same precision as the one used for the BDOF can be used. For example, 1/32 sample, 1/24 sample, or 1/64 sample may be predefined for MV_{block}.

[0247] In some embodiments, the gradients of prediction *Pred* may be calculated with a 3-tap filter [-1, 0, 1] as follows:

$$g_x(i,j) = I(i+1,j) - I(i-1,j)$$
$$g_y(i,j) = I(i,j+1) - I(i,j-1), \quad \text{(Eq. 23)}$$

where I(i, j) indicates the block prediction *Pred* at location (i, j).

[0248] For calculation of the gradients, in one embodiment, to reduce the memory bandwidth and complexity, the pixels on the extended borders may be copied from the nearest integer pixel position in the reference picture. Therefore, additional interpolation for the padding region is avoided. In another embodiment, the extended prediction samples may be generated by additional interpolation from the corresponding reference samples. In one embodiment, the precision of gradient is based on the bit-depth of source video. In another

embodiment, Sobel filter may be used to calculate the gradients for x and y directions. In another embodiment, Scharr filter may be used to calculate the gradients for x and y directions. In another embodiment, Prewitt filter may be used to calculate the gradients for x and y directions. In another embodiment, Roberts filter may be used to calculate the gradients for x and y directions. FIG. 26 shows examples of kernels of the Sobel, Scharr, Prewitt and Roberts filters.

[0249] In an embodiment, when storing MVs for a current picture, the MVs with a higher precision (higher than the highest precision supported by interpolation filter) may be stored and may be used in spatial MV prediction. In an embodiment, when storing MVs of the current picture into a frame buffer (to be used as TMVP for other pictures), the MVs with the higher precision may be rounded to the highest precision supported by the interpolation filter before MV compression.

[0250] In some embodiments, the PROF alike refinement process may not be used together with some coding modes, such as BDOF, DMVR, triangle merge, intra-inter mode, or the like.

[0251] VI. Examples of PROF or PROF Alike Refinement Processes

[0252] 1. Example PROF process

[0253] FIG. 27 shows a flowchart outlining a process (2700) according to an embodiment of the disclosure. The process (2700) can be used in the reconstruction of a block coded in an affine prediction mode (e.g., affine merge mode, affine AMVP, and the like), so to generate a prediction block for the block under reconstruction. In various embodiments, the process (2700) are executed by processing circuitry, such as the processing circuitry in the terminal devices (210), (220), (230) and (240), the processing circuitry that performs functions of the video decoder (310), the processing circuitry that performs functions of the video decoder (410), and the like. In some embodiments, the process (2700) is implemented in software instructions, thus when the processing circuitry executes the software instructions, the processing circuitry performs the process (2700). The process starts at (S2701) and proceeds to (S2710).

[0254] At S2710, a syntax element indicating whether a PROF for affine prediction is enabled. In an example, the syntax element is signaled at a sequence level, a slice level, a tile level, a tile group level, a picture level, or the like. In another example, the PROF for affine prediction can be enabled by default without signaling of the syntax element.

[0255] At S2720, it is determined whether to apply the PROF to an affine coded block. In an example, whether to apply the PROF to the affine coded block is determined

[0284] While this disclosure has described several exemplary embodiments, there are alterations, permutations, and various substitute equivalents, which fall within the scope of the disclosure.

WHAT IS CLAIMED IS:

1. A method of video coding by a video decoder, comprising:

receiving a picture level syntax element indicating whether a prediction refinement with optical flow (PROF) for affine prediction is disabled for a current picture;

determining whether to apply the PROF to an affine coded block in the current picture based on the picture level syntax element indicating whether the PROF for affine prediction is disabled for the current picture; and

wherein responsive to the picture level syntax element indicating the PROF for affine prediction is disabled for the current picture, the PROF for affine prediction is not applied to the affine coded block in the current picture, and

wherein responsive to the picture level syntax element indicating the PROF for affine prediction is not disabled for the current picture, the PROF for affine prediction is applied to the affine coded block.

2. The method of claim 1, further comprising:

responsive to the PROF for affine prediction being determined as applied to the affine coded block,

generating spatial gradients $g_x(i,j)$ and $g_y(i,j)$ at the sample location (i,j) in the affine coded block,

generating a prediction refinement $\Delta I(i,j)$ based on the spatial gradients $g_x(i,j)$ and $g_y(i,j)$, and

adding the prediction refinement $\Delta I(i,j)$ to the prediction sample $I(i,j)$ to generate a refined prediction sample.

3. The method of claim 1, wherein the PROF is enabled for affine prediction by default.

4. The method of claim 1, wherein the determining whether to apply the PROF for affine prediction to the affine coded block includes:

determining whether to apply the PROF for affine prediction to the affine coded block based on values of affine parameters of an affine model of the affine coded block.

5. The method of claim 1, wherein the generating the spatial gradients $g_x(i,j)$ and $g_y(i,j)$ at the sample location (i,j) includes:

generating the spatial gradients $g_x(i,j)$ and $g_y(i,j)$ at the sample location (i,j) based on a first prediction sample(s) of a first sub-block including the prediction sample $I(i,j)$ and a second prediction sample(s) of a second sub-block neighboring the first sub-block, the first sub-block and the second sub-block being partitioned from the affine coded block.

6. The method of claim 1, wherein the generating the spatial gradients $g_x(i,j)$ and $g_y(i,j)$ at the sample location (i,j) includes:

performing inter prediction for sub-blocks of the affine coded block; and

generating spatial gradients at sample locations on a basis of prediction samples of the entire affine coded block.

7. The method of claim 1, wherein the generating the spatial gradients $g_x(i,j)$ and $g_y(i,j)$ at the sample location (i,j) includes:

generating the spatial gradients $g_x(i,j)$ and $g_y(i,j)$ at the sample location (i,j) using a generated gradient filter on reference samples in a reference picture of the affine coded block.

8. The method of claim 7, wherein the generated gradient filter is generated by a convolution of a first gradient filter and an interpolation filter,

wherein application of the interpolation filter on the reference samples in the reference picture of the affine coded block generates prediction samples of the affine coded block, and subsequent application of the first gradient filter on the generated prediction samples of the affine coded block generates the spatial gradients $g_x(i,j)$ and $g_y(i,j)$.

9. The method of claim 1, wherein the determining whether to apply the PROF for affine prediction to the affine coded block includes:

determining whether to apply the PROF for affine prediction to the affine coded block based on the received picture level syntax element indicating whether the PROF for affine prediction is disabled, and a signaling indicating whether a local illumination compensation (LIC) is enabled.

10. The method of claim 1, wherein the affine coded block is coded in an affine merge mode, and an LIC flag value of the affine coded block is inherited from a neighboring block of the affine coded block that is used as a source for affine model inheritance or affine model construction of the affine coded block.

11. An apparatus, comprising circuitry configured to perform the method according to any one of claims 1 to 10.