



(51) International Patent Classification:

H04N 19/51 (2014.01)

(21) International Application Number:

PCT/CN2022/097557

(22) International Filing Date:

08 June 2022 (08.06.2022)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

PCT/CN2021/099529

10 June 2021 (10.06.2021)

CN

(71) Applicants: **BEIJING BYTEDANCE NETWORK**

TECHNOLOGY CO., LTD. [CN/CN]; Room B-0035, 2/F, No.3 Building, No. 30, Shixing Road, Shijingshan District, Beijing 100041 (CN). **BYTEDANCE INC.** [US/US]; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US).

(72) Inventors: **WANG, Yang**; Jinritoutiao Post Office, China Satellite Communications Tower, No.63, Zhichun Road, Haidian District, Beijing 100080 (CN). **ZHANG, Li**; 12655 West Jefferson Boulevard Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). **HE, Yuwen**; 12655 West Jefferson Boulevard Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). **ZHANG, Kai**; 12655 West Jef-

erson Boulevard Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US).

(74) Agent: **SHIHUI PARTNERS**; 42/F, Tower C, Beijing Yintai Centre, No. 2 Jianguomenwai Avenue, Chaoyang District, Beijing 100022 (CN).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

(54) Title: METHOD, DEVICE, AND MEDIUM FOR VIDEO PROCESSING

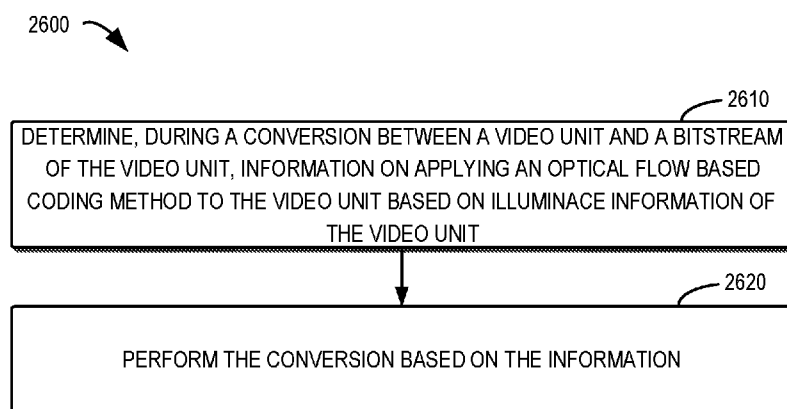


Fig. 26

(57) Abstract: Embodiments of the present disclosure provide a solution for video processing is proposed. A method for video processing is proposed. The method comprises: determining, during a conversion between a video unit and a bitstream of the video unit, information on applying an optical flow based coding method to the video unit based on illuminance information of the video unit; and performing the conversion based on the information. Compared with the conventional solution, the proposed method can advantageously improve the coding efficiency and performance.

Published:

— *with international search report (Art. 21(3))*

METHOD, DEVICE, AND MEDIUM FOR VIDEO PROCESSING

FIELD

[0001] Embodiments of the present disclosure relates generally to video coding techniques, and more particularly, to optical flow based coding.

BACKGROUND

[0002] In nowadays, digital video capabilities are being applied in various aspects of peoples' lives. Multiple types of video compression technologies, such as MPEG-2, MPEG-4, ITU-TH.263, ITU-TH.264/MPEG-4 Part 10 Advanced Video Coding (AVC), ITU-TH.265 high efficiency video coding (HEVC) standard, versatile video coding (VVC) standard, have been proposed for video encoding/decoding. However, coding efficiency of conventional video coding techniques is generally very low, which is undesirable.

SUMMARY

[0003] Embodiments of the present disclosure provide a solution for video processing.

[0004] In a first aspect, a method for video processing is proposed. The method comprises: determining, during a conversion between a video unit and a bitstream of the video unit, whether an optical flow based coding method is applied to the video unit based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit; and performing the conversion based on the determination. The method in accordance with the first aspect of the present disclosure considers illuminance information when determining whether to or how to apply an optical flow based coding method, which can advantageously improve the coding efficiency and performance.

[0005] In a second aspect, an apparatus for processing video data is proposed. The apparatus comprises a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to perform a method in accordance with the first aspect of the present disclosure.

[0006] In a third aspect, a non-transitory computer-readable storage medium is proposed. The non-transitory computer-readable storage medium stores instructions that cause a processor to perform a method in accordance with the first aspect of the present disclosure.

[0007] In a fourth aspect, a non-transitory computer-readable recording medium is proposed. The non-transitory computer-readable recording medium stores a bitstream of a video which is generated by a method performed by a video processing apparatus. The method comprises: determining whether an optical flow based coding method is applied to a video unit based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit; and generating a bitstream of the video unit based on the information.

[0008] In a fifth aspect, a method for storing a bitstream of a video is proposed. The method comprises: determining whether an optical flow based coding method is applied to a video unit based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit; generating a bitstream of the video unit based on the determination; and storing the bitstream in a non-transitory computer-readable recording medium.

[0009] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] Through the following detailed description with reference to the accompanying drawings, the above and other objectives, features, and advantages of example embodiments of the present disclosure will become more apparent. In the example embodiments of the present disclosure, the same reference numerals usually refer to the same components.

[0011] Fig. 1 illustrates a block diagram that illustrates an example video coding system, in accordance with some embodiments of the present disclosure;

[0012] Fig. 2 illustrates a block diagram that illustrates a first example video encoder, in accordance with some embodiments of the present disclosure;

[0013] Fig. 3 illustrates a block diagram that illustrates an example video decoder, in accordance with some embodiments of the present disclosure;

[0014] Fig. 4 is an example of encoder block diagram;

[0015] Fig. 5 is a schematic diagram of intra prediction modes;

- [0016] Fig. 6 illustrates a block diagram of reference samples for wide-angular intra prediction;
- [0017] Fig. 7 illustrates a block diagram of discontinuity in case of directions beyond 45 degree;
- [0018] Fig. 8 illustrates a block diagram of extended coding unit (CU) region used in bi-directional optical flow (BDOF);
- [0019] Fig. 9 shows control point based affine motion model;
- [0020] Fig. 10 shows affine MVF per subblock;
- [0021] Fig. 11 illustrates a block diagram of locations of inherited affine motion predictors;
- [0022] Fig. 12 illustrates a block diagram of control point motion vector inheritance;
- [0023] Fig. 13 illustrates a block diagram of locations of candidates position for constructed affine merge mode;
- [0024] Fig. 14 is an illustration of motion vector usage for proposed combined method;
- [0025] Fig. 15 shows subblock MV V_{SB} and pixel $\Delta v(l, j)$;
- [0026] Fig. 16 illustrates a block diagram of local illumination compensation;
- [0027] Fig. 17 shows no subsampling for the short side;
- [0028] Fig. 18 illustrates a block diagram of decoding side motion vector refinement;
- [0029] Fig. 19 illustrates a block diagram of diamond regions in the search area;
- [0030] Fig. 20 illustrates a block diagram of positions of spatial merge candidate;
- [0031] Fig. 21 illustrates a block diagram of candidate pairs for redundancy check of spatial merge candidate;
- [0032] Fig. 22 is an illustration of motion vector scaling for temporal merge candidate;
- [0033] Fig. 23 illustrates a block diagram of candidate positions for temporal merge candidate, C0 and C1;
- [0034] Fig. 24 shows a VVC spatial neighboring blocks of the current block;
- [0035] Fig. 25 illustrates a virtual block in the i-th search round;

[0036] Fig. 26 illustrates a flowchart of a method 2600 for video processing in accordance with some embodiments of the present disclosure;

[0037] Fig. 27 illustrates a flowchart of a method 2700 for video processing in accordance with some embodiments of the present disclosure;

[0038] Fig. 28 illustrates a flowchart of a method 2800 for video processing in accordance with some embodiments of the present disclosure;

[0039] Fig. 29 illustrates a flowchart of a method 2900 for video processing in accordance with some embodiments of the present disclosure; and

[0040] Fig. 30 illustrates a block diagram of a computing device in which various embodiments of the present disclosure can be implemented.

[0041] Throughout the drawings, the same or similar reference numerals usually refer to the same or similar elements.

DETAILED DESCRIPTION

[0042] Principle of the present disclosure will now be described with reference to some embodiments. It is to be understood that these embodiments are described only for the purpose of illustration and help those skilled in the art to understand and implement the present disclosure, without suggesting any limitation as to the scope of the disclosure. The disclosure described herein can be implemented in various manners other than the ones described below.

[0043] In the following description and claims, unless defined otherwise, all technical and scientific terms used herein have the same meaning as commonly understood by one of ordinary skills in the art to which this disclosure belongs.

[0044] References in the present disclosure to “one embodiment,” “an embodiment,” “an example embodiment,” and the like indicate that the embodiment described may include a particular feature, structure, or characteristic, but it is not necessary that every embodiment includes the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an example embodiment, it is submitted that it is within the knowledge of one skilled in the art to affect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0045] It shall be understood that although the terms “first” and “second” etc. may be used herein to describe various elements, these elements should not be limited by these terms. These terms are only used to distinguish one element from another. For example, a first element could be termed a second element, and similarly, a second element could be termed a first element, without departing from the scope of example embodiments. As used herein, the term “and/or” includes any and all combinations of one or more of the listed terms.

[0046] The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of example embodiments. As used herein, the singular forms “a”, “an” and “the” are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms “comprises”, “comprising”, “has”, “having”, “includes” and/or “including”, when used herein, specify the presence of stated features, elements, and/or components etc., but do not preclude the presence or addition of one or more other features, elements, components and/or combinations thereof.

Example Environment

[0047] Fig. 1 is a block diagram that illustrates an example video coding system 100 that may utilize the techniques of this disclosure. As shown, the video coding system 100 may include a source device 110 and a destination device 120. The source device 110 can be also referred to as a video encoding device, and the destination device 120 can be also referred to as a video decoding device. In operation, the source device 110 can be configured to generate encoded video data and the destination device 120 can be configured to decode the encoded video data generated by the source device 110. The source device 110 may include a video source 112, a video encoder 114, and an input/output (I/O) interface 116.

[0048] The video source 112 may include a source such as a video capture device. Examples of the video capture device include, but are not limited to, an interface to receive video data from a video content provider, a computer graphics system for generating video data, and/or a combination thereof.

[0049] The video data may comprise one or more pictures. The video encoder 114 encodes the video data from the video source 112 to generate a bitstream. The bitstream may include a sequence of bits that form a coded representation of the video data. The bitstream may include coded pictures and associated data. The coded picture is a coded representation of a picture. The associated data may include sequence parameter sets, picture parameter

sets, and other syntax structures. The I/O interface 116 may include a modulator/demodulator and/or a transmitter. The encoded video data may be transmitted directly to destination device 120 via the I/O interface 116 through the network 130A. The encoded video data may also be stored onto a storage medium/server 130B for access by destination device 120.

[0050] The destination device 120 may include an I/O interface 126, a video decoder 124, and a display device 122. The I/O interface 126 may include a receiver and/or a modem. The I/O interface 126 may acquire encoded video data from the source device 110 or the storage medium/server 130B. The video decoder 124 may decode the encoded video data. The display device 122 may display the decoded video data to a user. The display device 122 may be integrated with the destination device 120, or may be external to the destination device 120 which is configured to interface with an external display device.

[0051] The video encoder 114 and the video decoder 124 may operate according to a video compression standard, such as the High Efficiency Video Coding (HEVC) standard, Versatile Video Coding (VVC) standard and other current and/or further standards.

[0052] Fig. 2 is a block diagram illustrating an example of a video encoder 200, which may be an example of the video encoder 114 in the system 100 illustrated in Fig. 1, in accordance with some embodiments of the present disclosure.

[0053] The video encoder 200 may be configured to implement any or all of the techniques of this disclosure. In the example of Fig. 2, the video encoder 200 includes a plurality of functional components. The techniques described in this disclosure may be shared among the various components of the video encoder 200. In some examples, a processor may be configured to perform any or all of the techniques described in this disclosure.

[0054] In some embodiments, the video encoder 200 may include a partition unit 201, a predication unit 202 which may include a mode select unit 203, a motion estimation unit 204, a motion compensation unit 205 and an intra-prediction unit 206, a residual generation unit 207, a transform unit 208, a quantization unit 209, an inverse quantization unit 210, an inverse transform unit 211, a reconstruction unit 212, a buffer 213, and an entropy encoding unit 214.

[0055] In other examples, the video encoder 200 may include more, fewer, or different functional components. In an example, the predication unit 202 may include an intra block copy (IBC) unit. The IBC unit may perform predication in an IBC mode in which at least one reference picture is a picture where the current video block is located.

[0056] Furthermore, although some components, such as the motion estimation unit 204 and the motion compensation unit 205, may be integrated, but are represented in the example of Fig. 2 separately for purposes of explanation.

[0057] The partition unit 201 may partition a picture into one or more video blocks. The video encoder 200 and the video decoder 300 may support various video block sizes.

[0058] The mode select unit 203 may select one of the coding modes, intra or inter, e.g., based on error results, and provide the resulting intra-coded or inter-coded block to a residual generation unit 207 to generate residual block data and to a reconstruction unit 212 to reconstruct the encoded block for use as a reference picture. In some examples, the mode select unit 203 may select a combination of intra and inter predication (CIIP) mode in which the predication is based on an inter predication signal and an intra predication signal. The mode select unit 203 may also select a resolution for a motion vector (e.g., a sub-pixel or integer pixel precision) for the block in the case of inter-predication.

[0059] To perform inter prediction on a current video block, the motion estimation unit 204 may generate motion information for the current video block by comparing one or more reference frames from buffer 213 to the current video block. The motion compensation unit 205 may determine a predicted video block for the current video block based on the motion information and decoded samples of pictures from the buffer 213 other than the picture associated with the current video block.

[0060] The motion estimation unit 204 and the motion compensation unit 205 may perform different operations for a current video block, for example, depending on whether the current video block is in an I-slice, a P-slice, or a B-slice. As used herein, an “I-slice” may refer to a portion of a picture composed of macroblocks, all of which are based upon macroblocks within the same picture. Further, as used herein, in some aspects, “P-slices” and “B-slices” may refer to portions of a picture composed of macroblocks that are not dependent on macroblocks in the same picture.

[0061] In some examples, the motion estimation unit 204 may perform uni-directional prediction for the current video block, and the motion estimation unit 204 may search reference pictures of list 0 or list 1 for a reference video block for the current video block. The motion estimation unit 204 may then generate a reference index that indicates the reference picture in list 0 or list 1 that contains the reference video block and a motion vector that indicates a spatial displacement between the current video block and the reference video

block. The motion estimation unit 204 may output the reference index, a prediction direction indicator, and the motion vector as the motion information of the current video block. The motion compensation unit 205 may generate the predicted video block of the current video block based on the reference video block indicated by the motion information of the current video block.

[0062] Alternatively, in other examples, the motion estimation unit 204 may perform bi-directional prediction for the current video block. The motion estimation unit 204 may search the reference pictures in list 0 for a reference video block for the current video block and may also search the reference pictures in list 1 for another reference video block for the current video block. The motion estimation unit 204 may then generate reference indexes that indicate the reference pictures in list 0 and list 1 containing the reference video blocks and motion vectors that indicate spatial displacements between the reference video blocks and the current video block. The motion estimation unit 204 may output the reference indexes and the motion vectors of the current video block as the motion information of the current video block. The motion compensation unit 205 may generate the predicted video block of the current video block based on the reference video blocks indicated by the motion information of the current video block.

[0063] In some examples, the motion estimation unit 204 may output a full set of motion information for decoding processing of a decoder. Alternatively, in some embodiments, the motion estimation unit 204 may signal the motion information of the current video block with reference to the motion information of another video block. For example, the motion estimation unit 204 may determine that the motion information of the current video block is sufficiently similar to the motion information of a neighboring video block.

[0064] In one example, the motion estimation unit 204 may indicate, in a syntax structure associated with the current video block, a value that indicates to the video decoder 300 that the current video block has the same motion information as the another video block.

[0065] In another example, the motion estimation unit 204 may identify, in a syntax structure associated with the current video block, another video block and a motion vector difference (MVD). The motion vector difference indicates a difference between the motion vector of the current video block and the motion vector of the indicated video block. The video decoder 300 may use the motion vector of the indicated video block and the motion vector difference to determine the motion vector of the current video block.

[0066] As discussed above, video encoder 200 may predictively signal the motion vector. Two examples of predictive signaling techniques that may be implemented by video encoder 200 include advanced motion vector predication (AMVP) and merge mode signaling.

[0067] The intra prediction unit 206 may perform intra prediction on the current video block. When the intra prediction unit 206 performs intra prediction on the current video block, the intra prediction unit 206 may generate prediction data for the current video block based on decoded samples of other video blocks in the same picture. The prediction data for the current video block may include a predicted video block and various syntax elements.

[0068] The residual generation unit 207 may generate residual data for the current video block by subtracting (e.g., indicated by the minus sign) the predicted video block (s) of the current video block from the current video block. The residual data of the current video block may include residual video blocks that correspond to different sample components of the samples in the current video block.

[0069] In other examples, there may be no residual data for the current video block for the current video block, for example in a skip mode, and the residual generation unit 207 may not perform the subtracting operation.

[0070] The transform processing unit 208 may generate one or more transform coefficient video blocks for the current video block by applying one or more transforms to a residual video block associated with the current video block.

[0071] After the transform processing unit 208 generates a transform coefficient video block associated with the current video block, the quantization unit 209 may quantize the transform coefficient video block associated with the current video block based on one or more quantization parameter (QP) values associated with the current video block.

[0072] The inverse quantization unit 210 and the inverse transform unit 211 may apply inverse quantization and inverse transforms to the transform coefficient video block, respectively, to reconstruct a residual video block from the transform coefficient video block. The reconstruction unit 212 may add the reconstructed residual video block to corresponding samples from one or more predicted video blocks generated by the predication unit 202 to produce a reconstructed video block associated with the current video block for storage in the buffer 213.

[0073] After the reconstruction unit 212 reconstructs the video block, loop filtering operation may be performed to reduce video blocking artifacts in the video block.

[0074] The entropy encoding unit 214 may receive data from other functional components of the video encoder 200. When the entropy encoding unit 214 receives the data, the entropy encoding unit 214 may perform one or more entropy encoding operations to generate entropy encoded data and output a bitstream that includes the entropy encoded data.

[0075] Fig. 3 is a block diagram illustrating an example of a video decoder 300, which may be an example of the video decoder 124 in the system 100 illustrated in Fig. 1, in accordance with some embodiments of the present disclosure.

[0076] The video decoder 300 may be configured to perform any or all of the techniques of this disclosure. In the example of Fig. 3, the video decoder 300 includes a plurality of functional components. The techniques described in this disclosure may be shared among the various components of the video decoder 300. In some examples, a processor may be configured to perform any or all of the techniques described in this disclosure.

[0077] In the example of Fig. 3, the video decoder 300 includes an entropy decoding unit 301, a motion compensation unit 302, an intra prediction unit 303, an inverse quantization unit 304, an inverse transformation unit 305, and a reconstruction unit 306 and a buffer 307. The video decoder 300 may, in some examples, perform a decoding pass generally reciprocal to the encoding pass described with respect to video encoder 200.

[0078] The entropy decoding unit 301 may retrieve an encoded bitstream. The encoded bitstream may include entropy coded video data (e.g., encoded blocks of video data). The entropy decoding unit 301 may decode the entropy coded video data, and from the entropy decoded video data, the motion compensation unit 302 may determine motion information including motion vectors, motion vector precision, reference picture list indexes, and other motion information. The motion compensation unit 302 may, for example, determine such information by performing the AMVP and merge mode. AMVP is used, including derivation of several most probable candidates based on data from adjacent PBs and the reference picture. Motion information typically includes the horizontal and vertical motion vector displacement values, one or two reference picture indices, and, in the case of prediction regions in B slices, an identification of which reference picture list is associated with each index. As used herein, in some aspects, a “merge mode” may refer to deriving the motion information from spatially or temporally neighboring blocks.

[0079] The motion compensation unit 302 may produce motion compensated blocks, possibly performing interpolation based on interpolation filters. Identifiers for interpolation filters to be used with sub-pixel precision may be included in the syntax elements.

[0080] The motion compensation unit 302 may use the interpolation filters as used by the video encoder 200 during encoding of the video block to calculate interpolated values for sub-integer pixels of a reference block. The motion compensation unit 302 may determine the interpolation filters used by the video encoder 200 according to the received syntax information and use the interpolation filters to produce predictive blocks.

[0081] The motion compensation unit 302 may use at least part of the syntax information to determine sizes of blocks used to encode frame(s) and/or slice(s) of the encoded video sequence, partition information that describes how each macroblock of a picture of the encoded video sequence is partitioned, modes indicating how each partition is encoded, one or more reference frames (and reference frame lists) for each inter-encoded block, and other information to decode the encoded video sequence. As used herein, in some aspects, a “slice” may refer to a data structure that can be decoded independently from other slices of the same picture, in terms of entropy coding, signal prediction, and residual signal reconstruction. A slice can either be an entire picture or a region of a picture.

[0082] The intra prediction unit 303 may use intra prediction modes for example received in the bitstream to form a prediction block from spatially adjacent blocks. The inverse quantization unit 304 inverse quantizes, i.e., de-quantizes, the quantized video block coefficients provided in the bitstream and decoded by entropy decoding unit 301. The inverse transform unit 305 applies an inverse transform.

[0083] The reconstruction unit 306 may obtain the decoded blocks, e.g., by summing the residual blocks with the corresponding prediction blocks generated by the motion compensation unit 302 or intra-prediction unit 303. 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 the buffer 307, which provides reference blocks for subsequent motion compensation/intra prediction and also produces decoded video for presentation on a display device.

[0084] Some exemplary embodiments of the present disclosure will be described in detailed hereinafter. It should be understood that section headings are used in the present document to facilitate ease of understanding and do not limit the embodiments disclosed in a

section to only that section. Furthermore, while certain embodiments are described with reference to Versatile Video Coding or other specific video codecs, the disclosed techniques are applicable to other video coding technologies also. Furthermore, while some embodiments describe video coding steps in detail, it will be understood that corresponding steps decoding that undo the coding will be implemented by a decoder. Furthermore, the term video processing encompasses video coding or compression, video decoding or decompression and video transcoding in which video pixels are represented from one compressed format into another compressed format or at a different compressed bitrate.

1. Summary

The present disclosure is related to video coding technologies. Specifically, it is related optical flow based coding methods considering illuminance change, how to and/or whether to apply an optical flow based coding method depends on illuminance information, and other coding tools in image/video coding. It may be applied to the existing video coding standard like HEVC, or Versatile Video Coding (VVC). It may be also applicable to future video coding standards or video codec.

2. Background

Video coding standards have evolved primarily through the development of the well-known ITU-T and ISO/IEC standards. The ITU-T produced H.261 and H.263, ISO/IEC produced MPEG-1 and MPEG-4 Visual, and the two organizations jointly produced the H.262/MPEG-2 Video and H.264/MPEG-4 Advanced Video Coding (AVC) and H.265/HEVC standards. Since H.262, the video coding standards are based on the hybrid video coding structure wherein temporal prediction plus transform coding are utilized. To explore the future video coding technologies beyond HEVC, Joint Video Exploration Team (JVET) was founded by VCEG and MPEG jointly in 2015. Since then, many new methods have been adopted by JVET and put into the reference software named Joint Exploration Model (JEM). In April 2018, the Joint Video Expert Team (JVET) between VCEG (Q6/16) and ISO/IEC JTC1 SC29/WG11 (MPEG) was created to work on the VVC standard targeting at 50% bitrate reduction compared to HEVC.

The latest version of VVC draft, i.e., Versatile Video Coding (Draft 10) could be found at: http://phenix.it-sudparis.eu/jvet/doc_end_user/documents/20_Teleconference/wg11/JVET-T2001-v1.zip

The latest reference software of VVC, named VTM, could be found at:

https://vcgit.hhi.fraunhofer.de/jvet/VVCSOFTWARE_VTM/-/tags/VTM-11.0

2.1. Coding flow of a typical video codec

Fig. 4 shows an example of encoder block diagram of VVC, which contains three in-loop filtering blocks: deblocking filter (DF), sample adaptive offset (SAO) and ALF. Unlike DF, which uses predefined filters, SAO and ALF utilize the original samples of the current picture to reduce the mean square errors between the original samples and the reconstructed samples by adding an offset and by applying a finite impulse response (FIR) filter, respectively, with coded side information signalling the offsets and filter coefficients. ALF is located at the last processing stage of each picture and can be regarded as a tool trying to catch and fix artifacts created by the previous stages.

2.2. Intra mode coding with 67 intra prediction modes

To capture the arbitrary edge directions presented in natural video, the number of directional intra modes is extended from 33, as used in HEVC, to 65, as shown in Fig. 5, and the planar and DC modes remain the same. These denser directional intra prediction modes apply for all block sizes and for both luma and chroma intra predictions.

In the HEVC, every intra-coded block has a square shape and the length of each of its side is a power of 2. Thus, no division operations are required to generate an intra-predictor using DC mode. In VVC, blocks can have a rectangular shape that necessitates the use of a division operation per block in the general case. To avoid division operations for DC prediction, only the longer side is used to compute the average for non-square blocks.

2.2.1. Wide angle intra prediction

Although 67 modes are defined in the VVC, the exact prediction direction for a given intra prediction mode index is further dependent on the block shape. Conventional angular intra prediction directions are defined from 45 degrees to -135 degrees in clockwise direction. In VVC, several conventional angular intra prediction modes are adaptively replaced with wide-angle intra prediction modes for non-square blocks. The replaced modes are signalled using the original mode indexes, which are remapped to the indexes of wide angular modes after parsing. The total number of intra prediction modes is unchanged, i.e., 67, and the intra mode coding method is unchanged.

To support these prediction directions, the top reference with length $2W+1$, and the left reference with length $2H+1$, are defined as shown in Fig. 6.

The number of replaced modes in wide-angular direction mode depends on the aspect ratio of a block. The replaced intra prediction modes are illustrated in Table 1.

Table 1 Intra prediction modes replaced by wide-angular modes

Aspect ratio	Replaced intra prediction modes
$W / H == 16$	Modes 2,3,4,5,6,7,8,9,10,11,12, 13,14,15
$W / H == 8$	Modes 2,3,4,5,6,7,8,9,10,11,12, 13
$W / H == 4$	Modes 2,3,4,5,6,7,8,9,10,11
$W / H == 2$	Modes 2,3,4,5,6,7,8,9
$W / H == 1$	None
$W / H == 1/2$	Modes 59,60,61,62,63,64,65,66
$W / H == 1/4$	Mode 57,58,59,60,61,62,63,64,65,66
$W / H == 1/8$	Modes 55, 56,57,58,59,60,61,62,63,64,65,66
$W / H == 1/16$	Modes 53, 54, 55, 56,57,58,59,60,61,62,63,64,65,66

Fig. 7 illustrates a block diagram of discontinuity in case of directions beyond 45 degree. As shown in the diagram 700 of Fig. 7, two vertically adjacent predicted samples may use two non-adjacent reference samples in the case of wide-angle intra prediction. Hence, low-pass reference samples filter and side smoothing are applied to the wide-angle prediction to reduce the negative effect of the increased gap Δp_a . If a wide-angle mode represents a non-fractional offset. There are 8 modes in the wide-angle modes satisfy this condition, which are $[-14, -12, -10, -6, 72, 76, 78, 80]$. When a block is predicted by these modes, the samples in the reference buffer are directly copied without applying any interpolation. With this modification, the number of samples needed to be smoothing is reduced. Besides, it aligns the design of non-fractional modes in the conventional prediction modes and wide-angle modes.

In VVC, 4:2:2 and 4:4:4 chroma formats are supported as well as 4:2:0. Chroma derived mode (DM) derivation table for 4:2:2 chroma format was initially ported from HEVC extending the number of entries from 35 to 67 to align with the extension of intra prediction modes. Since HEVC specification does not support prediction angle below -135 degree and

above 45 degree, luma intra prediction modes ranging from 2 to 5 are mapped to 2. Therefore, chroma DM derivation table for 4:2:2: chroma format is updated by replacing some values of the entries of the mapping table to convert prediction angle more precisely for chroma blocks.

2.3. Inter prediction

For each inter-predicted CU, motion parameters consisting of motion vectors, reference picture indices and reference picture list usage index, and additional information needed for the new coding feature of VVC to be used for inter-predicted sample generation. The motion parameter can be signalled in an explicit or implicit manner. When a CU is coded with skip mode, the CU is associated with one PU and has no significant residual coefficients, no coded motion vector delta or reference picture index. A merge mode is specified whereby the motion parameters for the current CU are obtained from neighbouring CUs, including spatial and temporal candidates, and additional schedules introduced in VVC. The merge mode can be applied to any inter-predicted CU, not only for skip mode. The alternative to merge mode is the explicit transmission of motion parameters, where motion vector, corresponding reference picture index for each reference picture list and reference picture list usage flag and other needed information are signalled explicitly per each CU.

2.4. Intra block copy (IBC)

Intra block copy (IBC) is a tool adopted in HEVC extensions on SCC. It is well known that it significantly improves the coding efficiency of screen content materials. Since IBC mode is implemented as a block level coding mode, block matching (BM) is performed at the encoder to find the optimal block vector (or motion vector) for each CU. Here, a block vector is used to indicate the displacement from the current block to a reference block, which is already reconstructed inside the current picture. The luma block vector of an IBC-coded CU is in integer precision. The chroma block vector rounds to integer precision as well. When combined with AMVR, the IBC mode can switch between 1-pel and 4-pel motion vector precisions. An IBC-coded CU is treated as the third prediction mode other than intra or inter

prediction modes. The IBC mode is applicable to the CUs with both width and height smaller than or equal to 64 luma samples.

At the encoder side, hash-based motion estimation is performed for IBC. The encoder performs RD check for blocks with either width or height no larger than 16 luma samples. For non-merge mode, the block vector search is performed using hash-based search first. If hash search does not return valid candidate, block matching based local search will be performed. In the hash-based search, hash key matching (32-bit CRC) between the current block and a reference block is extended to all allowed block sizes. The hash key calculation for every position in the current picture is based on 4×4 sub-blocks. For the current block of a larger size, a hash key is determined to match that of the reference block when all the hash keys of all 4×4 sub-blocks match the hash keys in the corresponding reference locations. If hash keys of multiple reference blocks are found to match that of the current block, the block vector costs of each matched reference are calculated and the one with the minimum cost is selected. In block matching search, the search range is set to cover both the previous and current CTUs. At CU level, IBC mode is signalled with a flag and it can be signalled as IBC AMVP mode or IBC skip/merge mode as follows:

- IBC skip/merge mode: a merge candidate index is used to indicate which of the block vectors in the list from neighbouring candidate IBC coded blocks is used to predict the current block. The merge list consists of spatial, HMVP, and pairwise candidates.
- IBC AMVP mode: block vector difference is coded in the same way as a motion vector difference. The block vector prediction method uses two candidates as predictors, one from left neighbour and one from above neighbour (if IBC coded). When either neighbour is not available, a default block vector will be used as a predictor. A flag is signalled to indicate the block vector predictor index.

2.5. Bi-directional optical flow (BDOF)

The bi-directional optical flow (BDOF) tool is included in VVC. BDOF, previously referred to as BIO, was included in the JEM. Compared to the JEM version, the BDOF in VVC is a

simpler version that requires much less computation, especially in terms of number of multiplications and the size of the multiplier.

BDOF is used to refine the bi-prediction signal of a CU at the 4×4 subblock level. BDOF is applied to a CU if it satisfies all the following conditions:

- The CU is coded using “true” bi-prediction mode, i.e., one of the two reference pictures is prior to the current picture in display order and the other is after the current picture in display order
- The distances (i.e. POC difference) from two reference pictures to the current picture are same
- Both reference pictures are short-term reference pictures.
- The CU is not coded using affine mode or the SbTMVP merge mode
- CU has more than 64 luma samples
- Both CU height and CU width are larger than or equal to 8 luma samples
- BCW weight index indicates equal weight
- WP is not enabled for the current CU
- CIIP mode is not used for the current CU

BDOF is only applied to the luma component. As its name indicates, the BDOF mode is based on the optical flow concept, which assumes that the motion of an object is smooth. For each 4×4 subblock, a motion refinement (v_x, v_y) is calculated by minimizing the difference between the L0 and L1 prediction samples. The motion refinement is then used to adjust the bi-predicted sample values in the 4×4 subblock. The following steps are applied in the BDOF process.

First, the horizontal and vertical gradients, $\frac{\partial I^{(k)}}{\partial x}(i, j)$ and $\frac{\partial I^{(k)}}{\partial y}(i, j)$, $k = 0, 1$, of the two prediction signals are computed by directly calculating the difference between two neighboring samples, i.e.,

$$\begin{aligned}\frac{\partial I^{(k)}}{\partial x}(i, j) &= ((I^{(k)}(i+1, j) \gg \text{shift1}) - (I^{(k)}(i-1, j) \gg \text{shift1})) \\ \frac{\partial I^{(k)}}{\partial y}(i, j) &= ((I^{(k)}(i, j+1) \gg \text{shift1}) - (I^{(k)}(i, j-1) \gg \text{shift1}))\end{aligned}\tag{2-1}$$

where $I^{(k)}(i, j)$ are the sample value at coordinate (i, j) of the prediction signal in list k , $k = 0, 1$, and shift1 is calculated based on the luma bit depth, bitDepth , as $\text{shift1} = \max(6, \text{bitDepth} - 6)$.

Then, the auto- and cross-correlation of the gradients, S_1, S_2, S_3, S_5 and S_6 , are calculated as

$$\begin{aligned} S_1 &= \sum_{(i,j) \in \Omega} \text{Abs}(\psi_x(i, j)), \quad S_3 = \sum_{(i,j) \in \Omega} \theta(i, j) \cdot \text{Sign}(\psi_x(i, j)) \\ S_2 &= \sum_{(i,j) \in \Omega} \psi_x(i, j) \cdot \text{Sign}(\psi_y(i, j)) \\ S_5 &= \sum_{(i,j) \in \Omega} \text{Abs}(\psi_y(i, j)), \quad S_6 = \sum_{(i,j) \in \Omega} \theta(i, j) \cdot \text{Sign}(\psi_y(i, j)) \end{aligned} \quad (2-2)$$

where

$$\begin{aligned} \psi_x(i, j) &= \left(\frac{\partial I^{(1)}}{\partial x}(i, j) + \frac{\partial I^{(0)}}{\partial x}(i, j) \right) \gg n_a \\ \psi_y(i, j) &= \left(\frac{\partial I^{(1)}}{\partial y}(i, j) + \frac{\partial I^{(0)}}{\partial y}(i, j) \right) \gg n_a \\ \theta(i, j) &= (I^{(1)}(i, j) \gg n_b) - (I^{(0)}(i, j) \gg n_b) \end{aligned} \quad (2-3)$$

where Ω is a 6×6 window around the 4×4 subblock, and the values of n_a and n_b are set equal to $\min(1, \text{bitDepth} - 11)$ and $\min(4, \text{bitDepth} - 8)$, respectively.

The motion refinement (v_x, v_y) is then derived using the cross- and auto-correlation terms using the following:

$$\begin{aligned} v_x &= S_1 > 0 ? \text{clip3} \left(-th'_{BIO}, th'_{BIO}, -((S_3 \cdot 2^{n_b - n_a}) \gg \lfloor \log_2 S_1 \rfloor) \right) : 0 \\ v_y &= S_5 > 0 ? \text{clip3} \left(-th'_{BIO}, th'_{BIO}, - \left((S_6 \cdot 2^{n_b - n_a} \right. \right. \\ &\quad \left. \left. - ((v_x S_{2,m}) \ll n_{S_2} + v_x S_{2,s}) / 2 \right) \gg \lfloor \log_2 S_5 \rfloor \right) : 0 \end{aligned} \quad (2-4)$$

where $S_{2,m} = S_2 \gg n_{S_2}$, $S_{2,s} = S_2 \& (2^{n_{S_2}} - 1)$, $th'_{BIO} = 2^{\max(5, BD-7)}$. $\lfloor \cdot \rfloor$ is the floor function, and $n_{S_2} = 12$.

Based on the motion refinement and the gradients, the following adjustment is calculated for each sample in the 4×4 subblock:

$$b(x, y) = rnd \left(\left(v_x \left(\frac{\partial I^{(1)}(x, y)}{\partial x} - \frac{\partial I^{(0)}(x, y)}{\partial x} \right) + v_y \left(\frac{\partial I^{(1)}(x, y)}{\partial y} - \frac{\partial I^{(0)}(x, y)}{\partial y} \right) + 1 \right) / 2 \right) \quad (2-5)$$

Finally, the BDOF samples of the CU are calculated by adjusting the bi-prediction samples as follows:

$$pred_{BDOF}(x, y) = (I^{(0)}(x, y) + I^{(1)}(x, y) + b(x, y) + o_{offset}) \gg shift \quad (2-6)$$

These values are selected such that the multipliers in the BDOF process do not exceed 15-bit, and the maximum bit-width of the intermediate parameters in the BDOF process is kept within 32-bit.

In order to derive the gradient values, some prediction samples $I^{(k)}(i, j)$ in list k ($k = 0, 1$) outside of the current CU boundaries need to be generated. Fig. 8 illustrates a schematic diagram of extended CU region used in BDOF. As depicted in the diagram 800 of Fig. 8, the BDOF in VVC uses one extended row/column around the CU's boundaries. In order to control the computational complexity of generating the out-of-boundary prediction samples, prediction samples in the extended area (denoted as 810 in Fig. 8) are generated by taking the reference samples at the nearby integer positions (using floor() operation on the coordinates) directly without interpolation, and the normal 8-tap motion compensation interpolation filter is used to generate prediction samples within the CU (denoted as 820 in Fig. 8). These extended sample values are used in gradient calculation only. For the remaining steps in the BDOF process, if any sample and gradient values outside of the CU boundaries are needed, they are padded (i.e. repeated) from their nearest neighbors.

When the width and/or height of a CU are larger than 16 luma samples, it will be split into subblocks with width and/or height equal to 16 luma samples, and the subblock boundaries are treated as the CU boundaries in the BDOF process. The maximum unit size for BDOF process is limited to 16x16. For each subblock, the BDOF process could be skipped. When the SAD of between the initial L0 and L1 prediction samples is smaller than a threshold, the BDOF process is not applied to the subblock. The threshold is set equal to $(8 * W * (H > 1))$,

where W indicates the subblock width, and H indicates subblock height. To avoid the additional complexity of SAD calculation, the SAD between the initial L0 and L1 prediction samples calculated in DVMR process is re-used here.

If BCW is enabled for the current block, i.e., the BCW weight index indicates unequal weight, then bi-directional optical flow is disabled. Similarly, if WP is enabled for the current block, i.e., the luma_weight_lx_flag is 1 for either of the two reference pictures, then BDOF is also disabled. When a CU is coded with symmetric MVD mode or CIIP mode, BDOF is also disabled.

2.6. Affine motion compensated prediction

In HEVC, only translation motion model is applied for motion compensation prediction (MCP). While in the real world, there are many kinds of motion, e.g. zoom in/out, rotation, perspective motions and the other irregular motions. In VVC, a block-based affine transform motion compensation prediction is applied. As shown Fig. 9, the affine motion field of the block is described by motion information of two control point (4-parameter) or three control point motion vectors (6-parameter).

For 4-parameter affine motion model 910 in Fig. 9, motion vector at sample location (x, y) in a block is derived as:

$$\begin{cases} mv_x = \frac{mv_{1x}-mv_{0x}}{W}x + \frac{mv_{1y}-mv_{0y}}{W}y + mv_{0x} \\ mv_y = \frac{mv_{1y}-mv_{0y}}{W}x + \frac{mv_{1x}-mv_{0x}}{W}y + mv_{0y} \end{cases} \quad (2-7)$$

For 6-parameter affine motion model 920 in Fig. 9, motion vector at sample location (x, y) in a block is derived as:

$$\begin{cases} mv_x = \frac{mv_{1x}-mv_{0x}}{W}x + \frac{mv_{2x}-mv_{0x}}{H}y + mv_{0x} \\ mv_y = \frac{mv_{1y}-mv_{0y}}{W}x + \frac{mv_{2y}-mv_{0y}}{H}y + mv_{0y} \end{cases} \quad (2-8)$$

Where (mv_{0x}, mv_{0y}) is 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 order to simplify the motion compensation prediction, block based affine transform prediction is applied. Fig. 10 illustrates a schematic diagram 1000 of affine MVF per subblock. To derive motion vector of each 4×4 luma subblock, the motion vector of the center sample of each subblock, as shown in Fig. 10, is calculated according to above equations, and rounded to $1/16$ fraction accuracy. Then the motion compensation interpolation filters are applied to generate the prediction of each subblock with derived motion vector. The subblock size of chroma-components is also set to be 4×4 . The MV of a 4×4 chroma subblock is calculated as the average of the MVs of the four corresponding 4×4 luma subblocks.

As done for translational motion inter prediction, there are also two affine motion inter prediction modes: affine merge mode and affine AMVP mode.

2.6.1. Affine merge prediction

AF_MERGE mode can be applied for CUs with both width and height larger than or equal to 8. In this mode the CPMVs of the current CU is generated based on the motion information of the spatial neighbouring CUs. . There can be up to five CPMVP candidates and an index is signalled to indicate the one to be used for the current CU. The following three types of CPVM candidate are used to form the affine merge candidate list:

- Inherited affine merge candidates that extrapolated from the CPMVs of the neighbour CUs
- Constructed affine merge candidates CPMVPs that are derived using the translational MVs of the neighbour CUs
- Zero MVs

In VVC, there are maximum two inherited affine candidates, which are derived from affine motion model of the neighbouring blocks, one from left neighbouring CUs and one from above neighbouring CUs. Fig. 11 illustrates a schematic diagram 1100 of locations of inherited affine motion predictors. The candidate blocks are shown in Fig. 11. For the left predictor, the scan order is $A0 \rightarrow A1$, and for the above predictor, the scan order is $B0 \rightarrow B1 \rightarrow B2$. Only the first inherited candidate from each side is selected. No pruning check is performed between two inherited candidates. When a neighbouring affine CU is identified, its control point motion vectors are used to derive the CPMVP candidate in the affine merge list of the current CU. Fig. 12 illustrates a schematic diagram 1200 of control point motion vector inheritance. As shown in 12, if the neighbour left bottom block A 1210 is coded in affine mode, the

motion vectors v_2 , v_3 and v_4 of the top left corner, above right corner and left bottom corner of the CU 1220 which contains the block A 1210 are attained. When block A 1210 is coded with 4-parameter affine model, the two CPMVs of the current CU are calculated according to v_2 , and v_3 . In case that block A is coded with 6-parameter affine model, the three CPMVs of the current CU are calculated according to v_2 , v_3 and v_4 .

Constructed affine candidate means the candidate is constructed by combining the neighbour translational motion information of each control point. The motion information for the control points is derived from the specified spatial neighbours and temporal neighbour shown in Fig. 13 which illustrates a schematic diagram 1300 of locations of candidates position for constructed affine merge mode. $CPMV_k$ ($k=1, 2, 3, 4$) represents the k -th control point. For $CPMV_1$, the B2->B3->A2 blocks are checked and the MV of the first available block is used. For $CPMV_2$, the B1->B0 blocks are checked and for $CPMV_3$, the A1->A0 blocks are checked. For TMVP is used as $CPMV_4$ if it's available.

After MVs of four control points are attained, affine merge candidates are constructed based on that motion information. The following combinations of control point MVs are used to construct in order:

$\{CPMV_1, CPMV_2, CPMV_3\}$, $\{CPMV_1, CPMV_2, CPMV_4\}$, $\{CPMV_1, CPMV_3, CPMV_4\}$,
 $\{CPMV_2, CPMV_3, CPMV_4\}$, $\{CPMV_1, CPMV_2\}$, $\{CPMV_1, CPMV_3\}$

The combination of 3 CPMVs constructs a 6-parameter affine merge candidate and the combination of 2 CPMVs constructs a 4-parameter affine merge candidate. To avoid motion scaling process, if the reference indices of control points are different, the related combination of control point MVs is discarded.

After inherited affine merge candidates and constructed affine merge candidate are checked, if the list is still not full, zero MVs are inserted to the end of the list.

2.6.2. Affine AMVP prediction

Affine AMVP mode can be applied for CUs with both width and height larger than or equal to 16. An affine flag in CU level is signalled in the bitstream to indicate whether affine AMVP mode is used and then another flag is signalled to indicate whether 4-parameter affine or 6-

parameter affine. In this mode, the difference of the CPMVs of current CU and their predictors CPMVPs is signalled in the bitstream. The affine AMVP candidate list size is 2 and it is generated by using the following four types of CPMV candidate in order:

- Inherited affine AMVP candidates that extrapolated from the CPMVs of the neighbour CUs
- Constructed affine AMVP candidates CPMVPs that are derived using the translational MVs of the neighbour CUs
- Translational MVs from neighbouring CUs
- Zero MVs

The checking order of inherited affine AMVP candidates is same to the checking order of inherited affine merge candidates. The only difference is that, for AMVP candidate, only the affine CU that has the same reference picture as in current block is considered. No pruning process is applied when inserting an inherited affine motion predictor into the candidate list.

Constructed AMVP candidate is derived from the specified spatial neighbours shown in Fig. 13. The same checking order is used as done in affine merge candidate construction. In addition, reference picture index of the neighbouring block is also checked. The first block in the checking order that is inter coded and has the same reference picture as in current CUs is used. There is only one When the current CU is coded with 4-parameter affine mode, and mv_0 and mv_1 are both available, they are added as one candidate in the affine AMVP list. When the current CU is coded with 6-parameter affine mode, and all three CPMVs are available, they are added as one candidate in the affine AMVP list. Otherwise, constructed AMVP candidate is set as unavailable.

If affine AMVP list candidates is still less than 2 after inherited affine AMVP candidates and Constructed AMVP candidate are checked, mv_0 , mv_1 , and mv_2 will be added, in order, as the translational MVs to predict all control point MVs of the current CU, when available. Finally, zero MVs are used to fill the affine AMVP list if it is still not full.

2.6.3. Affine motion information storage

In VVC, the CPMVs of affine CUs are stored in a separate buffer. The stored CPMVs are only used to generate the inherited CPMVPs in affine merge mode and affine AMVP mode for the lately coded CUs. The subblock MVs derived from CPMVs are used for motion compensation, MV derivation of merge/AMVP list of translational MVs and de-blocking.

To avoid the picture line buffer for the additional CPMVs, affine motion data inheritance from the CUs from above CTU is treated differently to the inheritance from the normal neighbouring CUs. If the candidate CU for affine motion data inheritance is in the above CTU line, the bottom-left and bottom-right subblock MVs in the line buffer instead of the CPMVs are used for the affine MVP derivation. In this way, the CPMVs are only stored in local buffer. If the candidate CU is 6-parameter affine coded, the affine model is degraded to 4-parameter model. As shown in Fig. 14, along the top CTU boundary, the bottom-left and bottom right subblock motion vectors of a CU are used for affine inheritance of the CUs in bottom CTUs.

2.6.4. Prediction refinement with optical flow for affine mode

Subblock based affine motion compensation can save memory access bandwidth and reduce computation complexity compared to pixel-based motion compensation, at the cost of prediction accuracy penalty. To achieve a finer granularity of motion compensation, prediction refinement with optical flow (PROF) is used to refine the subblock based affine motion compensated prediction without increasing the memory access bandwidth for motion compensation. In VVC, after the subblock based affine motion compensation is performed, luma prediction sample is refined by adding a difference derived by the optical flow equation. The PROF is described as following four steps:

Step 1) The subblock-based affine motion compensation is performed to generate subblock prediction $I(i, j)$.

Step2) The spatial gradients $g_x(i, j)$ and $g_y(i, j)$ of the subblock prediction are calculated at each sample location using a 3-tap filter $[-1, 0, 1]$. The gradient calculation is exactly the same as gradient calculation in BDOF.

$$g_x(i, j) = (I(i + 1, j) \gg \text{shift1}) - (I(i - 1, j) \gg \text{shift1}) \quad (2-9)$$

$$g_y(i, j) = (I(i, j + 1) \gg \text{shift1}) - (I(i, j - 1) \gg \text{shift1}) \quad (2-10)$$

shift1 is used to control the gradient's precision. The subblock (i.e. 4x4) prediction is extended by one sample on each side for the gradient calculation. To avoid additional memory bandwidth and additional interpolation computation, those extended samples on the extended borders are copied from the nearest integer pixel position in the reference picture.

Step 3) The luma prediction refinement is calculated by the following optical flow equation.

$$\Delta I(i, j) = g_x(i, j) * \Delta v_x(i, j) + g_y(i, j) * \Delta v_y(i, j) \quad (2-11)$$

where the $\Delta v(i, j)$ is the difference between sample MV computed for sample location (i, j) , denoted by $v(i, j)$, and the subblock MV of the subblock to which sample (i, j) belongs, as shown in Fig. 15. The $\Delta v(i, j)$ is quantized in the unit of 1/32 luam sample precision.

Since the affine model parameters and the sample location relative to the subblock center are not changed from subblock to subblock, $\Delta v(i, j)$ can be calculated for the first subblock, and reused for other subblocks in the same CU. Let $dx(i, j)$ and $dy(i, j)$ be the horizontal and vertical offset from the sample location (i, j) to the center of the subblock (x_{SB}, y_{SB}) , $\Delta v(x, y)$ can be derived by the following equation,

$$\begin{cases} dx(i, j) = i - x_{SB} \\ dy(i, j) = j - y_{SB} \end{cases} \quad (2-12)$$

$$\begin{cases} \Delta v_x(i, j) = C * dx(i, j) + D * dy(i, j) \\ \Delta v_y(i, j) = E * dx(i, j) + F * dy(i, j) \end{cases} \quad (2-13)$$

In order to keep accuracy, the center of the subblock (x_{SB}, y_{SB}) is calculated as $((W_{SB} - 1)/2, (H_{SB} - 1)/2)$, where W_{SB} and H_{SB} are the subblock width and height, respectively.

For 4-parameter affine model,

$$\begin{cases} C = F = \frac{v_{1x} - v_{0x}}{w} \\ E = -D = \frac{v_{1y} - v_{0y}}{w} \end{cases} \quad (2-14)$$

For 6-parameter affine model,

$$\begin{cases} C = \frac{v_{1x}-v_{0x}}{w} \\ D = \frac{v_{2x}-v_{0x}}{h} \\ E = \frac{v_{1y}-v_{0y}}{w} \\ F = \frac{v_{2y}-v_{0y}}{h} \end{cases} \quad (2-15)$$

where (v_{0x}, v_{0y}) , (v_{1x}, v_{1y}) , (v_{2x}, v_{2y}) are the top-left, top-right and bottom-left control point motion vectors, w and h are the width and height of the CU.

Step 4) Finally, the luma prediction refinement $\Delta I(i, j)$ is added to the subblock prediction $I(i, j)$. The final prediction I' is generated as the following equation.

$$I'(i, j) = I(i, j) + \Delta I(i, j) \quad (2-16)$$

PROF is not be applied in two cases for an affine coded CU: 1) all control point MVs are the same, which indicates the CU only has translational motion; 2) the affine motion parameters are greater than a specified limit because the subblock based affine MC is degraded to CU based MC to avoid large memory access bandwidth requirement.

A fast encoding method is applied to reduce the encoding complexity of affine motion estimation with PROF. PROF is not applied at affine motion estimation stage in following two situations: a) if this CU is not the root block and its parent block does not select the affine mode as its best mode, PROF is not applied since the possibility for current CU to select the affine mode as best mode is low; b) if the magnitude of four affine parameters (C, D, E, F) are all smaller than a predefined threshold and the current picture is not a low delay picture, PROF is not applied because the improvement introduced by PROF is small for this case. In this way, the affine motion estimation with PROF can be accelerated.

2.7. Bi-prediction with CU-level weight (BCW)

In HEVC, the bi-prediction signal is generated by averaging two prediction signals obtained from two different reference pictures and/or using two different motion vectors. In VVC, the bi-prediction mode is extended beyond simple averaging to allow weighted averaging of the two prediction signals.

$$P_{\text{bi-pred}} = ((8 - w) * P_0 + w * P_1 + 4) \gg 3 \quad (2-17)$$

Five weights are allowed in the weighted averaging bi-prediction, $w \in \{-2, 3, 4, 5, 10\}$. For each bi-predicted CU, the weight w is determined in one of two ways: 1) for a non-merge CU, the weight w is determined in one of two ways: 1) for a non-merge CU, the weight index is signalled after the motion vector difference; 2) for a merge CU, the weight index is inferred from neighbouring blocks based on the merge candidate index. BCW is only applied to CUs with 256 or more luma samples (i.e., CU width times CU height is greater than or equal to 256). For low-delay pictures, all 5 weights are used. For non-low-delay pictures, only 3 weights ($w \in \{3, 4, 5\}$) are used.

- At the encoder, fast search algorithms are applied to find the weight index without significantly increasing the encoder complexity. These algorithms are summarized as follows. For further details readers are referred to the VTM software and document JVET-L0646. When combined with AMVR, unequal weights are only conditionally checked for 1-pel and 4-pel motion vector precisions if the current picture is a low-delay picture.
- When combined with affine, affine ME will be performed for unequal weights if and only if the affine mode is selected as the current best mode.
- When the two reference pictures in bi-prediction are the same, unequal weights are only conditionally checked.
- Unequal weights are not searched when certain conditions are met, depending on the POC distance between current picture and its reference pictures, the coding QP, and the temporal level.

The BCW weight index is coded using one context coded bin followed by bypass coded bins. The first context coded bin indicates if equal weight is used; and if unequal weight is used, additional bins are signalled using bypass coding to indicate which unequal weight is used.

Weighted prediction (WP) is a coding tool supported by the H.264/AVC and HEVC standards to efficiently code video content with fading. Support for WP was also added into the VVC standard. WP allows weighting parameters (weight and offset) to be signalled for each reference picture in each of the reference picture lists L0 and L1. Then, during motion compensation, the weight(s) and offset(s) of the corresponding reference picture(s) are applied. WP and BCW are designed for different types of video content. In order to avoid interactions between WP and BCW, which will complicate VVC decoder design, if a CU uses WP, then the BCW weight index is not signalled, and w is inferred to be 4 (i.e. equal weight is

applied). For a merge CU, the weight index is inferred from neighbouring blocks based on the merge candidate index. This can be applied to both normal merge mode and inherited affine merge mode. For constructed affine merge mode, the affine motion information is constructed based on the motion information of up to 3 blocks. The BCW index for a CU using the constructed affine merge mode is simply set equal to the BCW index of the first control point MV.

In VVC, CIIP and BCW cannot be jointly applied for a CU. When a CU is coded with CIIP mode, the BCW index of the current CU is set to 2, e.g., equal weight.

2.8. Local illumination compensation (LIC)

Local illumination compensation (LIC) is a coding tool to address the issue of local illumination changes between current picture and its temporal reference pictures. The LIC is based on a linear model where a scaling factor and an offset are applied to the reference samples to obtain the prediction samples of a current block. Specifically, the LIC can be mathematically modeled by the following equation:

$$P(x, y) = \alpha \cdot P_r(x + v_x, y + v_y) + \beta$$

where $P(x, y)$ is the prediction signal of the current block at the coordinate (x, y) ; $P_r(x + v_x, y + v_y)$ is the reference block pointed by the motion vector (v_x, v_y) ; α and β are the corresponding scaling factor and offset that are applied to the reference block. Fig. 16 illustrates the LIC process. In Fig. 16, when the LIC is applied for a block, a least mean square error (LMSE) method is employed to derive the values of the LIC parameters (i.e., α and β) by minimizing the difference between the neighboring samples of the current block (i.e., the template T in Fig. 16) and their corresponding reference samples in the temporal reference pictures (i.e., either $T0$ or $T1$ in Fig. 16). Additionally, to reduce the computational complexity, both the template samples and the reference template samples are subsampled

(adaptive subsampling) to derive the LIC parameters, i.e., only the shaded samples in Fig. 16 are used to derive α and β .

To improve the coding performance, no subsampling for the short side is performed as shown in Fig. 17.

2.9. Decoder side motion vector refinement (DMVR)

In order to increase the accuracy of the MVs of the merge mode, a bilateral-matching (BM) based decoder side motion vector refinement is applied in VVC. In bi-prediction operation, a refined MV is searched around the initial MVs in the reference picture list L0 and reference picture list L1. The BM method calculates the distortion between the two candidate blocks in the reference picture list L0 and list L1. Fig. 18 is a schematic diagram illustrating the decoding side motion vector refinement. As illustrated in Fig. 18, the SAD between the blocks 1810 and 1812 based on each MV candidate around the initial MV is calculated, where the block 1810 is in a reference picture 1801 in the list L0 and the block 1812 is in a reference picture 1803 in the List L1 for the current picture 1802. The MV candidate with the lowest SAD becomes the refined MV and used to generate the bi-predicted signal.

In VVC, the application of DMVR is restricted and is only applied for the CUs which are coded with following modes and features:

- CU level merge mode with bi-prediction MV
- One reference picture is in the past and another reference picture is in the future with respect to the current picture
- The distances (i.e. POC difference) from two reference pictures to the current picture are same
- Both reference pictures are short-term reference pictures
- CU has more than 64 luma samples
- Both CU height and CU width are larger than or equal to 8 luma samples
- BCW weight index indicates equal weight
- WP is not enabled for the current block
- CIIP mode is not used for the current block

The refined MV derived by DMVR process is used to generate the inter prediction samples and also used in temporal motion vector prediction for future pictures coding. While the original MV is used in deblocking process and also used in spatial motion vector prediction for future CU coding.

The additional features of DMVR are mentioned in the following sub-clauses.

2.9.1. Searching scheme

In DVMR, the search points are surrounding the initial MV and the MV offset obey the MV difference mirroring rule. In other words, any points that are checked by DMVR, denoted by candidate MV pair (MV0, MV1) obey the following two equations:

$$MV0' = MV0 + MV_offset \quad (2-18)$$

$$MV1' = MV1 - MV_offset \quad (2-19)$$

Where MV_offset represents the refinement offset between the initial MV and the refined MV in one of the reference pictures. The refinement search range is two integer luma samples from the initial MV. The searching includes the integer sample offset search stage and fractional sample refinement stage.

25 points full search is applied for integer sample offset searching. The SAD of the initial MV pair is first calculated. If the SAD of the initial MV pair is smaller than a threshold, the integer sample stage of DMVR is terminated. Otherwise SADs of the remaining 24 points are calculated and checked in raster scanning order. The point with the smallest SAD is selected as the output of integer sample offset searching stage. To reduce the penalty of the uncertainty of DMVR refinement, it is proposed to favor the original MV during the DMVR process. The SAD between the reference blocks referred by the initial MV candidates is decreased by 1/4 of the SAD value.

The integer sample search is followed by fractional sample refinement. To save the calculational complexity, the fractional sample refinement is derived by using parametric error surface equation, instead of additional search with SAD comparison. The fractional sample

refinement is conditionally invoked based on the output of the integer sample search stage. When the integer sample search stage is terminated with center having the smallest SAD in either the first iteration or the second iteration search, the fractional sample refinement is further applied.

In parametric error surface based sub-pixel offsets estimation, the center position cost and the costs at four neighboring positions from the center are used to fit a 2-D parabolic error surface equation of the following form

$$E(x, y) = A(x - x_{min})^2 + B(y - y_{min})^2 + C \quad (2-20)$$

where (x_{min}, y_{min}) corresponds to the fractional position with the least cost and C corresponds to the minimum cost value. By solving the above equations by using the cost value of the five search points, the (x_{min}, y_{min}) is computed as:

$$x_{min} = (E(-1,0) - E(1,0)) / (2(E(-1,0) + E(1,0) - 2E(0,0))) \quad (2-21)$$

$$y_{min} = (E(0,-1) - E(0,1)) / (2(E(0,-1) + E(0,1) - 2E(0,0))) \quad (2-22)$$

The value of x_{min} and y_{min} are automatically constrained to be between -8 and 8 since all cost values are positive and the smallest value is $E(0,0)$. This corresponds to half pel offset with 1/16th-pel MV accuracy in VVC. The computed fractional (x_{min}, y_{min}) are added to the integer distance refinement MV to get the sub-pixel accurate refinement delta MV.

2.9.2. Bilinear-interpolation and sample padding

In VVC, the resolution of the MVs is 1/16 luma samples. The samples at the fractional position are interpolated using an 8-tap interpolation filter. In DMVR, the search points are surrounding the initial fractional-pel MV with integer sample offset, therefore the samples of those fractional position need to be interpolated for DMVR search process. To reduce the calculation complexity, the bi-linear interpolation filter is used to generate the fractional samples for the searching process in DMVR. Another important effect is that by using bi-linear filter is that with 2-sample search range, the DMVR does not access more reference samples compared to the normal motion compensation process. After the refined MV is attained with DMVR search process, the normal 8-tap interpolation filter is applied to generate the final

prediction. In order to not access more reference samples to normal MC process, the samples, which is not needed for the interpolation process based on the original MV but is needed for the interpolation process based on the refined MV, will be padded from those available samples.

2.9.3. Maximum DMVR processing unit

When the width and/or height of a CU are larger than 16 luma samples, it will be further split into subblocks with width and/or height equal to 16 luma samples. The maximum unit size for DMVR searching process is limit to 16x16.

2.10. Multi-pass decoder-side motion vector refinement

In this contribution, a multi-pass decoder-side motion vector refinement is applied instead of DMVR. In the first pass, bilateral matching (BM) is applied to a coding block. In the second pass, BM is applied to each 16x16 subblock within the coding block. In the third pass, MV in each 8x8 subblock is refined by applying bi-directional optical flow (BDOF). The refined MVs are stored for both spatial and temporal motion vector prediction.

2.10.1. First pass – Block based bilateral matching MV refinement

In the first pass, a refined MV is derived by applying BM to a coding block. Similar to decoder-side motion vector refinement (DMVR), the refined MV is searched around the two initial MVs (MV0 and MV1) in the reference picture lists L0 and L1. The refined MVs (MV0_pass1 and MV1_pass1) are derived around the initiate MVs based on the minimum bilateral matching cost between the two reference blocks in L0 and L1.

BM performs local search to derive integer sample precision intDeltaMV and half-pel sample precision halfDeltaMv . The local search applies a 3x3 square search pattern to loop through the search range $[-sHor, sHor]$ in a horizontal direction and $[-sVer, sVer]$ in a vertical direction, wherein, the values of $sHor$ and $sVer$ are determined by the block dimension, and the maximum value of $sHor$ and $sVer$ is 8.

The bilateral matching cost is calculated as: $\text{bilCost} = \text{mvDistanceCost} + \text{sadCost}$. When the block size $\text{cbW} * \text{cbH}$ is greater than 64, MRSAD cost function is applied to remove the DC effect of the distortion between the reference blocks. When the bilCost at the center point of the 3×3 search pattern has the minimum cost, the intDeltaMV or halfDeltaMV local search is terminated. Otherwise, the current minimum cost search point becomes the new center point of the 3×3 search pattern and the search for the minimum cost continues, until it reaches the end of the search range.

The existing fractional sample refinement is further applied to derive the final deltaMV . The refined MVs after the first pass are then derived as:

- $\text{MV0_pass1} = \text{MV0} + \text{deltaMV}$
- $\text{MV1_pass1} = \text{MV1} - \text{deltaMV}$

2.10.2. Second pass – Subblock based bilateral matching MV refinement

In the second pass, a refined MV is derived by applying BM to a 16×16 grid subblock. For each subblock, the refined MV is searched around the two MVs (MV0_pass1 and MV1_pass1), obtained on the first pass for the reference picture list L0 and L1. The refined MVs ($\text{MV0_pass2}(\text{sbIdx2})$ and $\text{MV1_pass2}(\text{sbIdx2})$) are derived based on the minimum bilateral matching cost between the two reference subblocks in L0 and L1.

For each subblock, BM performs full search to derive integer sample precision intDeltaMV . The full search has a search range $[-\text{sHor}, \text{sHor}]$ in a horizontal direction and $[-\text{sVer}, \text{sVer}]$ in a vertical direction, wherein, the values of sHor and sVer are determined by the block dimension, and the maximum value of sHor and sVer is 8.

The bilateral matching cost is calculated by applying a cost factor to the SATD cost between the two reference subblocks, as: $\text{bilCost} = \text{satdCost} * \text{costFactor}$. The search area $(2 * \text{sHor} + 1) * (2 * \text{sVer} + 1)$ is divided up to 5 diamond shape search regions shown in in the diagram 1900 of Fig. 19. Each search region is assigned a costFactor , which is determined by the distance (intDeltaMV) between each search point and the starting MV, and each diamond region is processed in the order starting from the center of the search area. In each region, the search points are processed in the raster scan order starting from the top left going to the bottom right corner of the region. When the minimum bilCost within the current search

region is less than a threshold equal to $sbW * sbH$, the int-pel full search is terminated, otherwise, the int-pel full search continues to the next search region until all search points are examined.

BM performs local search to derive half sample precision halfDeltaMv. The search pattern and cost function are the same as defined in 2.9.1.

The existing VVC DMVR fractional sample refinement is further applied to derive the final $\text{deltaMV}(sbIdx2)$. The refined MVs at second pass is then derived as:

- $MV0_pass2(sbIdx2) = MV0_pass1 + \text{deltaMV}(sbIdx2)$
- $MV1_pass2(sbIdx2) = MV1_pass1 - \text{deltaMV}(sbIdx2)$

2.10.3. Third pass – Subblock based bi-directional optical flow MV refinement

In the third pass, a refined MV is derived by applying BDOF to an 8×8 grid subblock. For each 8×8 subblock, BDOF refinement is applied to derive scaled V_x and V_y without clipping starting from the refined MV of the parent subblock of the second pass. The derived $\text{bioMv}(V_x, V_y)$ is rounded to 1/16 sample precision and clipped between -32 and 32.

The refined MVs ($MV0_pass3(sbIdx3)$ and $MV1_pass3(sbIdx3)$) at third pass are derived as:

- $MV0_pass3(sbIdx3) = MV0_pass2(sbIdx2) + \text{bioMv}$
- $MV1_pass3(sbIdx3) = MV0_pass2(sbIdx2) - \text{bioMv}$

2.11. Sample-based BDOF

In the sample-based BDOF, instead of deriving motion refinement (V_x, V_y) on a block basis, it is performed per sample.

The coding block is divided into 8×8 subblocks. For each subblock, whether to apply BDOF or not is determined by checking the SAD between the two reference subblocks against a threshold. If decided to apply BDOF to a subblock, for every sample in the subblock, a sliding 5×5 window is used and the existing BDOF process is applied for every sliding window to derive V_x and V_y . The derived motion refinement (V_x, V_y) is applied to adjust the bi-predicted sample value for the center sample of the window.

2.12. Extended merge prediction

In VVC, the merge candidate list is constructed by including the following five types of candidates in order:

- (1) Spatial MVP from spatial neighbour CUs
- (2) Temporal MVP from collocated CUs
- (3) History-based MVP from a FIFO table
- (4) Pairwise average MVP
- (5) Zero MVs.

The size of merge list is signalled in sequence parameter set header and the maximum allowed size of merge list is 6. For each CU code in merge mode, an index of best merge candidate is encoded using truncated unary binarization (TU). The first bin of the merge index is coded with context and bypass coding is used for other bins.

The derivation process of each category of merge candidates is provided in this session. As done in HEVC, VVC also supports parallel derivation of the merging candidate lists for all CUs within a certain size of area.

2.12.1. Spatial candidates derivation

The derivation of spatial merge candidates in VVC is same to that in HEVC except the positions of first two merge candidates are swapped. Fig. 20 is a schematic diagram 2000 illustrating positions of a spatial merge candidate. A maximum of four merge candidates are selected among candidates located in the positions depicted in Fig. 20. The order of derivation is B_0 , A_0 , B_1 , A_1 and B_2 . Position B_2 is considered only when one or more than one CUs of position B_0 , A_0 , B_1 , A_1 are not available (e.g. because it belongs to another slice or tile) or is intra coded. After candidate at position A_1 is added, the addition of the remaining candidates is subject to a redundancy check which ensures that candidates with same motion information are excluded from the list so that coding efficiency is improved. To reduce computational complexity, not all possible candidate pairs are considered in the mentioned redundancy check. Fig. 21 is a schematic diagram 2100 illustrating candidate pairs considered for redundancy check of spatial merge candidates. Instead only the pairs linked with an arrow in Fig. 21 are considered and a candidate is only added to the list if the corresponding candidate used for redundancy check has not the same motion information.

2.12.2. Temporal candidates derivation

In this step, only one candidate is added to the list. Particularly, in the derivation of this temporal merge candidate, a scaled motion vector is derived based on co-located CU belonging to the collocated reference picture. The reference picture list to be used for derivation of the co-located CU is explicitly signalled in the slice header. The scaled motion vector for temporal merge candidate is obtained as illustrated by the dotted line in the diagram 2200 of Fig. 22, which is scaled from the motion vector of the co-located CU using the POC distances, tb and td , where tb is defined to be the POC difference between the reference picture of the current picture and the current picture and td is defined to be the POC difference between the reference picture of the co-located picture and the co-located picture. The reference picture index of temporal merge candidate is set equal to zero.

Fig. 23 is a schematic diagram 2300 illustrating candidate positions for temporal merge candidate, C_0 and C_1 . The position for the temporal candidate is selected between candidates C_0 and C_1 , as depicted in Fig. 23. If CU at position C_0 is not available, is intra coded, or is outside of the current row of CTUs, position C_1 is used. Otherwise, position C_0 is used in the derivation of the temporal merge candidate.

2.12.3. History-based merge candidates derivation

The history-based MVP (HMVP) merge candidates are added to merge list after the spatial MVP and TMVP. In this method, the motion information of a previously coded block is stored in a table and used as MVP for the current CU. The table with multiple HMVP candidates is maintained during the encoding/decoding process. The table is reset (emptied) when a new CTU row is encountered. Whenever there is a non-subblock inter-coded CU, the associated motion information is added to the last entry of the table as a new HMVP candidate.

The HMVP table size S is set to be 6, which indicates up to 6 History-based MVP (HMVP) candidates may be added to the table. When inserting a new motion candidate to the table, a constrained first-in-first-out (FIFO) rule is utilized wherein redundancy check is firstly applied to find whether there is an identical HMVP in the table. If found, the identical HMVP is removed from the table and all the HMVP candidates afterwards are moved forward.

HMVP candidates could be used in the merge candidate list construction process. The latest several HMVP candidates in the table are checked in order and inserted to the candidate list after the TMVP candidate. Redundancy check is applied on the HMVP candidates to the spatial or temporal merge candidate.

To reduce the number of redundancy check operations, the following simplifications are introduced:

Number of HMPV candidates is used for merge list generation is set as $(N \leq 4) ? M : (8 - N)$, wherein N indicates number of existing candidates in the merge list and M indicates number of available HMVP candidates in the table.

Once the total number of available merge candidates reaches the maximally allowed merge candidates minus 1, the merge candidate list construction process from HMVP is terminated.

2.12.4. Pair-wise average merge candidates derivation

Pairwise average candidates are generated by averaging predefined pairs of candidates in the existing merge candidate list, and the predefined pairs are defined as $\{(0, 1), (0, 2), (1, 2), (0, 3), (1, 3), (2, 3)\}$, where the numbers denote the merge indices to the merge candidate list. The averaged motion vectors are calculated separately for each reference list. If both motion vectors are available in one list, these two motion vectors are averaged even when they point to different reference pictures; if only one motion vector is available, use the one directly; if no motion vector is available, keep this list invalid.

When the merge list is not full after pair-wise average merge candidates are added, the zero MVPs are inserted in the end until the maximum merge candidate number is encountered.

2.12.5. Merge estimation region

Merge estimation region (MER) allows independent derivation of merge candidate list for the CUs in the same merge estimation region (MER). A candidate block that is within the same MER to the current CU is not included for the generation of the merge candidate list of the current CU. In addition, the updating process for the history-based motion vector

predictor candidate list is updated only if $(x_{Cb} + cbWidth) \gg \text{Log2ParMrgLevel}$ is greater than $x_{Cb} \gg \text{Log2ParMrgLevel}$ and $(y_{Cb} + cbHeight) \gg \text{Log2ParMrgLevel}$ is great than $(y_{Cb} \gg \text{Log2ParMrgLevel})$ and where (x_{Cb}, y_{Cb}) is the top-left luma sample position of the current CU in the picture and $(cbWidth, cbHeight)$ is the CU size. The MER size is selected at encoder side and signalled as `log2_parallel_merge_level_minus2` in the sequence parameter set.

2.13. New merge candidates

2.13.1. Non-adjacent merge candidates derivation

Fig. 24 illustrates a schematic diagram 2400 of VVC spatial neighboring blocks of the current block. In VVC, five spatially neighboring blocks shown in Fig. 24 as well as one temporal neighbor are used to derive merge candidates.

It is proposed to derive the additional merge candidates from the positions non-adjacent to the current block using the same pattern as that in VVC. To achieve this, for each search round i , a virtual block is generated based on the current block as follows:

First, the relative position of the virtual block to the current block is calculated by:

$$\text{Offsetx} = -i \times \text{gridX}, \text{Offsety} = -i \times \text{gridY}$$

where the Offsetx and Offsety denote the offset of the top-left corner of the virtual block relative to the top-left corner of the current block, gridX and gridY are the width and height of the search grid.

Second, the width and height of the virtual block are calculated by:

$$\text{newWidth} = i \times 2 \times \text{gridX} + \text{currWidth} \quad \text{newHeight} = i \times 2 \times \text{gridY} + \text{currHeight}.$$

where the currWidth and currHeight are the width and height of current block. The newWidth and newHeight are the width and height of new virtual block.

gridX and gridY are currently set to currWidth and currHeight , respectively.

Fig. 25 illustrates a schematic diagram of a virtual block in the i th search round, which shows the relationship between the virtual block and the current block.

After generating the virtual block, the blocks A_i , B_i , C_i , D_i and E_i can be regarded as the VVC spatial neighboring blocks of the virtual block and their positions are obtained with the same pattern as that in VVC. Obviously, the virtual block is the current block if the search round i is 0. In this case, the blocks A_i , B_i , C_i , D_i and E_i are the spatially neighboring blocks that are used in VVC merge mode.

When constructing the merge candidate list, the pruning is performed to guarantee each element in merge candidate list to be unique. The maximum search round is set to 1, which means that five non-adjacent spatial neighbor blocks are utilized.

Non-adjacent spatial merge candidates are inserted into the merge list after the temporal merge candidate in the order of $B_1 \rightarrow A_1 \rightarrow C_1 \rightarrow D_1 \rightarrow E_1$.

2.13.2. STMVP

It is proposed to derive an averaging candidate as STMVP candidate using three spatial merge candidates and one temporal merge candidate.

STMVP is inserted before the above-left spatial merge candidate.

The STMVP candidate is pruned with all the previous merge candidates in the merge list.

For the spatial candidates, the first three candidates in the current merge candidate list are used.

For the temporal candidate, the same position as VTM / HEVC collocated position is used.

For the spatial candidates, the first, second, and third candidates inserted in the current merge candidate list before STMVP are denoted as F , S , and, T .

The temporal candidate with the same position as VTM / HEVC collocated position used in TMVP is denoted as Col .

The motion vector of the STMVP candidate in prediction direction X (denoted as $mvLX$) is derived as follows:

- 1) If the reference indices of the four merge candidates are all valid and are all equal to zero in prediction direction X ($X = 0$ or 1),

$$mvLX = (mvLX_F + mvLX_S + mvLX_T + mvLX_Col) >> 2$$

- 2) If reference indices of three of the four merge candidates are valid and are equal to zero in prediction direction X ($X = 0$ or 1),

$$mvLX = (mvLX_F \times 3 + mvLX_S \times 3 + mvLX_Col \times 2) >> 3 \quad \text{or}$$

$$mvLX = (mvLX_F \times 3 + mvLX_T \times 3 + mvLX_Col \times 2) >> 3 \quad \text{or}$$

$$mvLX = (mvLX_S \times 3 + mvLX_T \times 3 + mvLX_Col \times 2) >> 3$$

- 3) If reference indices of two of the four merge candidates are valid and are equal to zero in prediction direction X (X = 0 or 1),

$$mvLX = (mvLX_F + mvLX_Col) >> 1 \text{ or}$$

$$mvLX = (mvLX_S + mvLX_Col) >> 1 \text{ or}$$

$$mvLX = (mvLX_T + mvLX_Col) >> 1$$

Note: If the temporal candidate is unavailable, the STMVP mode is off.

2.13.3. Merge list size

If considering both non-adjacent and STMVP merge candidates, the size of merge list is signalled in sequence parameter set header and the maximum allowed size of merge list is 8.

3. Problems

In current design of current optical flow based coding methods (e.g., bi-directional optical flow (BDOF) and prediction refinement with optical flow (PROF)), the illuminance change is not considered. How to deal with optical flow based coding methods when illuminance change occurs needs to be explored.

4. Embodiments of the present disclosure

The detailed embodiments below should be considered as examples to explain general concepts. These embodiments should not be interpreted in a narrow way. Furthermore, these embodiments can be combined in any manner.

On determination of optical flow based coding methods using illuminance information

1. Whether to apply an optical flow based coding method to a video unit may depend on whether illuminance change occurs.
 - a. In one example, the optical flow based coding method may not be applied to the video unit when illuminance change occurs.
 - i. In one example, how to detect the illuminance change may depend on neighbouring samples (adjacent or non-adjacent) of the video unit.
 - ii. In one example, whether the illuminance change of the video unit occurs may be indicated by a syntax element and signalled in the bit-stream.

- iii. In one example, whether the illuminance change of the video unit occurs may depend on whether and/or how a certain coding tool (such as LIC and BCW) is applied.
 - iv. In one example, the optical flow based coding method may be not applied to a sample/pixel of the video unit when there is change on the illuminance of the sample/pixel.
2. How to apply an optical flow based coding method to a video unit may depend on whether illuminance change occurs.
- a. In one example, the illuminance change may be included in the process of the optical flow based coding method.
 - i. In one example, a value may be subtracted when calculating the gradients in the process of the optical flow based coding method.
 - ii. In one example, when calculating the difference of a sample/pixel in a first prediction block and a sample/pixel in a second prediction block, a first value may be firstly subtracted from the sample/pixel in the first prediction block and a second value may be firstly subtracted from the sample/pixel in the second prediction block.
 - iii. Instead of using the first/second prediction blocks directly obtained from motion information, it is proposed to firstly revise the obtained prediction blocks via a function, i.e. $f(P_i)$ is used in the optical flow procedure, instead of the sample value P_i .
 - 1) In one example, the function is a linear function, (e.g., $f(P_i) = a \cdot P_i + b$) wherein a and b are the linear parameters, and P_i denotes a sample/pixel in the prediction blocks.
 - 2) In one example, a and/ b may be determined by coding tools such as LIC and BCW.
 - 3) Alternatively, furthermore, the linear parameter (a, b) may be derived using neighbouring samples/pixels.
 - 4) Alternatively, furthermore, the linear parameter (a, b) may be signalled.
 - 5) Alternatively, furthermore, the linear parameter sets (a, b) may be different for the two prediction blocks.
 - 6) Alternatively, furthermore, the obtained prediction blocks may be revised using a non-linear function, such as polynomial functions.
 - iv. The model parameters, such as linear model, of illumination change may be jointly optimized with the optical flow parameters jointly.
 - 1) The luminance change model parameters and optical flow parameters of one block may be solved with least square regression method iteratively.

3. The detection and/or calculation of illumination changes may be performed in a first level, and the decision of how to and/or whether to apply the optical flow based coding method may be done in a second level.
 - a. In one example, the 1st/2nd levels are both block level.
 - b. In one example, the 1st/2nd levels are both picture level.
 - c. In one example, the 1st/2nd levels are both sub-block level.
 - d. In one example, the 1st level is block level, and the 2nd level is sub-block level.
 - e. In one example, all samples/pixels in the first level may be utilized.
 - i. Alternatively, partial of them may be utilized.
4. For the block-level optical flow based coding method, the detection and/or calculation of illumination changes may involve more samples in addition to the prediction blocks of the current block.
5. For the subblock-level optical flow based coding method, the detection and/or calculation of illumination changes may involve more samples in addition to the prediction blocks of the current subblock in a block.
6. Whether to and/or how to apply the optical flow based coding method to the video unit may depend on whether and/or how a coding tool solving illumination changes (e.g., BCW, LIC) is applied to the video unit.
 - a. In one example, the coding tool may refer to a local illumination compensation method, and/or a bi-prediction with CU-level weight method, and/or affine compensation method.
 - b. In one example, the optical flow based coding method may not be applied when the coding tool is applied to the video unit.
 - c. In one example, the optical flow based coding method may be applied to the video unit when the coding tool is applied to the video unit.
7. In one example, whether to enable the optical flow based coding method for a video unit may depend on the illuminance information of the video unit and/or the reference video unit.
 - a. In one example, whether to enable the optical flow based coding method for a video unit may depend on the illumination information of two reference pictures.
 - i. In one example, if illumination changes occurs among the two reference pictures, it is disallowed to enable the optical flow based coding method.
 - ii. In one example, one of the two reference pictures is from list X, and the other is from list Y.

- iii. In one example, the absolute POC distance of the two reference pictures is equal to twice of the absolute POC distance of one reference picture relative to the current video unit.
 - iv. When calculating illumination information, all samples/pixels or partial of them in the two reference pictures may be utilized.
- b. In one example, the video unit may refer to picture/subpicture/tile/slice/coding tree unit/a group of coding tree units/coding unit.
- c. In one example, whether to enable the optical flow based coding method for a video unit may depend on the illumination information of current picture and one or more reference pictures.
 - i. In one example, the determination of whether illuminance change occurs depends on the current picture and/or one or more reference pictures.
 - 1) In one example, the original samples or reconstructed samples in the reference pictures may be used to determine whether illuminance change occurs.
 - 2) In one example, the original samples or partial reconstructed samples or prediction samples of current picture may be used to determine whether illuminance change occurs.
 - 3) In one example, histograms are calculated for one or more reference pictures, and it is determined illumination changes occurs when the difference of the histograms is larger than T.
 - a) In one example, T may be set adaptively dependent on the size of the current picture.
 - b) In one example, T may depend on the coding information.
 - c) In one example, T may depend on current picture.
 - d) In one example, T may be calculated using the histograms of current picture and the reference pictures.
 - ii. In one example, if illumination change occurs among current picture and the reference pictures, it is disallowed to enable the optical flow based coding method.
- d. In one example, the illuminance information may refer to the sample values of one or more components between the video unit and its reference video unit.
 - i. In one example, the component may refer to luma component.
 - ii. In one example, the component may refer to one or more chroma components.
 - iii. In one example, the video unit and/or the reference video unit may be a coding block, such as coding unit/prediction unit/transform unit.

- iv. In one example, a first feature of sample value is calculated for the video unit, and a second feature of sample value is calculated for the reference video unit. When the difference of the first feature and the second feature is larger than T, the optical flow based method may be not applied to the video unit.
 - 1) In one example, the first feature for the video unit may be calculated using neighbouring samples (adjacent or non-adjacent) of the video unit.
 - a) Alternatively, a prediction signal may be derived for the video unit (e.g., intra prediction), and the first feature of the video unit may be calculated using the prediction signal.
 - 2) In one example, the second feature may be calculated using the reconstructed samples of the reference unit.
 - 3) In one example, the feature may refer to the mean value, and/or variance value.
 - 4) In one example, the feature may refer to the histogram of sample values.
 - 5) In one example, the determination of T may depend on coding information.
 - a) In one example, the coding information may refer to the dimension, and/or size of the video unit.
- 8. In above examples, the optical flow based coding method may refer to a bi-directional optical flow method in which the optical flow is used to refine the bi-prediction signal of a coding block, and/or a prediction refinement with optical flow for affine mode in which the optical flow is used to refine the affine motion compensated prediction, and/or other coding methods in which the optical flow is used to generate/refine the prediction/reconstruction signal of a coding block.
 - a. In one example, it may be the PROF.
 - b. In one example, it may be the BDOF.
- 9. The term of “illuminance change” of a sample/pixel may refer to the sample/pixel value changes a lot between two different video units (e.g., current picture and its reference picture). For example, $d = \text{abs}(P1 - P2)$ where P1 and P2 denote two samples/pixels in two different video units, illuminance change occurs when d is larger than a certain value D.
- 10. The term of “illuminance change” of a video unit may refer to the values of most samples/pixels, and/or the mean value of samples/pixels in the video unit change a lot between two different video units.
 - a. For example, $d = \text{abs}(m1 - m2)$ where m1 and m2 denote the output of a function applied to two associated video units.

- i. In one example, the function is defined to be the mean values, e.g., two mean values of samples/pixels in two different video units are calculated as m_1 and m_2 , respectively.
 - ii. In one example, illuminance change occurs when d is larger than a certain value D .
- 11. In above examples, the variable D may be predefined.
 - a. Alternatively, D may be derived on-the-fly, e.g., according to decoded information (e.g., adjacent or non-adjacent samples/pixels in current picture/different pictures).
 - b. Alternatively, D may be signalled in a bitstream.
- 12. In above examples, the video unit may refer to the video unit may refer to colour component/sub-picture/slice/tile/coding tree unit (CTU)/CTU row/groups of CTU/coding unit (CU)/prediction unit (PU)/transform unit (TU)/ coding tree block (CTB)/ coding block (CB)/prediction block(PB)/transform block (TB)/a block/sub-block of a block/sub-region within a block/ any other region that contains more than one sample or pixel.
 - a. In one example, the determination of illuminance change/information may refer to luma component and/or chroma components.
- 13. Whether to and/or how to apply the disclosed methods above may be signalled at sequence level/group of pictures level/picture level/slice level/tile group level, such as in sequence header/picture header/SPS/VPS/DPS/DCI/PPS/APS/slice header/tile group header.
- 14. Whether to and/or how to apply the disclosed methods above may be signalled at PB/TB/CB/PU/TU/CU/VPDU/CTU/CTU row/slice/tile/sub-picture/other kinds of region contains more than one sample or pixel.
- 15. Whether to and/or how to apply the disclosed methods above may be dependent on coded information, such as block size, colour format, single/dual tree partitioning, colour component, slice/picture type.

[0085] Embodiments of the present disclosure are related to determining whether to and/or how to apply an optical flow based coding method to a video unit based on illuminance information.

[0086] As used herein, the term “video unit” used herein may refer to one or more of: a color component, a sub-picture, a slice, a tile, a coding tree unit (CTU), a CTU row, a group of CTUs, a coding unit (CU), a prediction unit (PU), a transform unit (TU), a coding tree block (CTB), a coding block (CB), a prediction block(PB), a transform block (TB), a block,

a sub-block of a block, a sub-region within the block, or a region that comprises more than one sample or pixel.

[0087] Fig. 26 illustrates a flowchart of a method 2600 for video processing in accordance with some embodiments of the present disclosure. The method 2600 may be implemented during a conversion between a video unit and a bitstream of the video unit.

[0088] At block 2610, during a conversion between a video unit and a bitstream of the video unit, information on applying an optical flow based coding method to the video unit is determined based on illuminance information of the video unit. According to embodiments of the present disclosure, the information on applying the optical flow based coding method to the video unit comprises whether to apply the optical flow based coding method to the video unit. According to embodiments of the present disclosure, the information on applying the optical flow based coding method to the video unit comprises how to apply the optical flow based coding method to the video unit.

[0089] At block 2620, the conversion is performed based on the information. In some embodiments, the conversion may comprise encoding the video unit into the bitstream. In some embodiments, the conversion may comprise decoding the video unit from the bitstream.

[0090] In some embodiments, a bitstream of a video may be stored in a non-transitory computer-readable recording medium. The bitstream of the video can be generated by a method performed by a video processing apparatus. According to the method, information on applying an optical flow based coding method to the video unit is determined based on illuminance information of the video unit, and a bitstream of the video unit is generated based on the information.

[0091] In some embodiments, information on applying an optical flow based coding method to the video unit is determined based on illuminance information of the video unit. A bitstream of the video unit is generated based on the information, and the bitstream is stored in a non-transitory computer-readable recording medium.

[0092] According to embodiments of the present disclosure, illuminance information can be considered when determining whether to or how to apply an optical flow based coding method. Compared with the conventional solution, some embodiments of the present disclosure can advantageously improve the coding efficiency and coding performance.

[0093] Fig. 27 illustrates a flowchart of a method 2700 for video processing in accordance with some embodiments of the present disclosure. The method 2700 may be implemented during a conversion between a video unit and a bitstream of the video unit.

[0094] At block 2710, during a conversion between a video unit and a bitstream of the video unit, whether an optical flow based coding method is applied to the video unit is determined based on illuminance information of the video unit.

[0095] At block 2720, the conversion is performed based on the determination. In some embodiments, the conversion may comprise encoding the video unit into the bitstream. In some embodiments, the conversion may comprise decoding the video unit from the bitstream.

[0096] According to embodiments of the present disclosure, illuminance information can be considered when determining whether to apply an optical flow based coding method. Compared with the conventional solution, some embodiments of the present disclosure can advantageously improve the coding efficiency and coding performance.

[0097] In some embodiments, whether to apply the optical flow based coding method to the video unit may depend on whether illuminance change occurs. In some embodiments, if the illuminance change occurs, the optical flow based coding method may not be applied to the video unit. In this way, it can ensure that the optical flow based coding method is applied in a proper scenario.

[0098] In some embodiments, whether the illuminance change of the video unit occurs may be determined based on a set of neighbor samples of the video unit. In some embodiments, the set of neighbor samples may be adjacent to the video unit. In some embodiments, the set of neighbor samples may be non-adjacent to the video unit.

[0099] In some embodiments, a syntax element in the bitstream may indicate whether the illuminance change occurs. In other words, whether the illuminance change of the video unit occurs may be indicated by a syntax element and signalled in the bitstream. In this way, it can reduce the computation burden at the decoder side when determining whether to apply the optical flow based coding method.

[00100] In some embodiments, whether the illuminance change of the video unit occurs may be determined based on whether a coding tool is applied to the video unit. In some embodiments, whether the illuminance change of the video unit occurs may be determined based on how the coding tool is applied to the video unit. In some embodiments, the coding

tool may refer to at least one of: a local illumination compensation method, a bi-prediction with CU-level weight method, or affine compensation method. For example, in some embodiments, the coding tool may comprise BCW. In some embodiments, the coding tool may comprise LIC. In some embodiments, if the coding tool is applied, it may mean that the illuminance is changed.

[00101] In some embodiments, if the illuminance change occurs on a sample or a pixel of the video unit, the optical flow based coding method may not be applied to the sample or the pixel of the video unit. For example, if the illuminance change occurs on left-top sample(s) or pixel(s), the optical flow based coding method may not be applied to the left-top sample(s) or pixel(s).

[00102] In some embodiments, a determination of an illuminance change of the video unit may be performed in a first level. In some embodiments, a determination of whether to and/or how to the optical flow based coding method may be performed in a second level. In some embodiments, the first level and the second level may be both block level. In some embodiments, the first level and the second level may be both picture level. In some embodiments, the first level and the second level may be both sub-block level. In some embodiments, the first level may be block level and the second level may be sub-block level. In some embodiments, all of samples or pixels in the first level may be utilized. In some embodiments, a part of the samples or pixels in the first level may be utilized.

[00103] In some embodiments, if the optical flow based coding method is block-level, at least one of detection or a calculation of an illuminance change of the video unit may involve samples in addition to a set of prediction blocks of a current block associated with the video unit. In some embodiments, if the optical flow based coding method is subblock-level, at least one of detection or a calculation of an illuminance change of the video unit may involve samples in addition to a set of prediction blocks of a current subblock associated with the video unit in a block.

[00104] In some embodiments, whether the optical flow based coding method is applied to the video unit may be determined based on at least one of: whether a coding tool is applied to the video unit, or how the coding tool is applied to the video unit. In some embodiments, the coding tool may refer to at least one of: a local illumination compensation method, a bi-prediction with CU-level weight method, or affine compensation method. For example, in some embodiments, the coding tool may comprise BCW. In some embodiments, if the the

coding tool is applied to the video unit, the optical flow based coding method may not be applied to the video unit. In some embodiments, if the coding tool is applied to the video unit, the optical flow based coding method may be applied to the video unit.

[00105] In some embodiments, the optical flow based coding method may refer to at least one of: a bi-directional optical flow method in which an optical flow is used to refine a bi-prediction signal of a coding block, a prediction refinement with optical flow for affine mode in which the optical flow is used to refine an affine motion compensated prediction, or a coding method in which the optical flow is used to generate or refine a prediction/reconstruction signal of a coding block. In some embodiments, the optical flow based coding method may be a bi-directional optical flow (BDOF). In some embodiments, the optical flow based coding method may be a prediction refinement with optical flow (PROF).

[00106] In some embodiments, the illuminance change of a sample/pixel may refer to the sample/pixel value changes a lot between two different video units (e.g., current picture and its reference picture). In some embodiments, if a change of sample or pixel values between two video units is larger than a first threshold value, an illuminance change may occur. For example, the change may be calculated by $d = \text{abs}(P1 - P2)$ where $P1$ and $P2$ denote two samples/pixels in two different video units and abs represents an absolute value operation. In this case, illuminance change may occur if d is larger than a certain value D . In some embodiments, the first threshold value may be predefined. In some embodiments, the first threshold value may be derived dynamically. For example, the first threshold value may be derived according to decoded information (e.g., adjacent or non-adjacent samples/pixels in current picture/different pictures). In some embodiments, the first threshold value may be indicated in the bitstream.

[00107] In some embodiments, the illuminance change of a video unit may refer to the values of most samples/pixels, and/or the mean value of samples/pixels in the video unit change a lot between two different video units. In some embodiments, if a change of sample or pixel values in the video unit between two video units is larger than a second threshold value, an illuminance change occurs. In some embodiments, if a change of mean values of sample or pixel values in the video unit between two video units is larger than the second threshold value, the illuminance change may occur. For example, the change may be calculated by $d = \text{abs}(m1 - m2)$, where $m1$ and $m2$ denote the output of a function applied to two associated video units and abs represents an absolute value operation. In some embodiments,

the the function may be defined to be mean values. For example, two mean values of samples/pixels in two different video units are calculated as m_1 and m_2 , respectively. In some embodiments, the second threshold value may be predefined. In some embodiments, the second threshold value may be derived dynamically. For example, the second threshold value may be derived according to decoded information (e.g., adjacent or non-adjacent samples/pixels in current picture/different pictures). In some embodiments, the second threshold value may be indicated in the bitstream.

[00108] In some embodiments, the illuminance information or an illuminance change of the video unit may comprise at least one of: a luma component, or a chroma component.

[00109] In some embodiments, an indication of whether to apply the optical flow based coding method may be indicated at one of the followings: sequence level, group of pictures level, picture level, slice level, or tile group level. For example, in some embodiments, the indication of whether to apply the optical flow based coding method may be indicated in one of the following: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a dependency parameter set (DPS), a decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter sets (APS), a slice header, or a tile group header.

[00110] In some embodiments, an indication of whether to apply the optical flow based coding method may be included in one of the following: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, a sub-picture, or a region containing more than one sample or pixel.

[00111] In some embodiments, whether the optical flow based coding method is applied may be determined based on coded information of the video unit. The coded information may comprise at least one of: a block size, a colour format, a single and/or dual tree partitioning, a colour component, a slice type, or a picture type.

[00112] In some embodiments, a bitstream of a video may be stored in a non-transitory computer-readable recording medium. The bitstream of the video can be generated by a method performed by a video processing apparatus. According to the method, whether an optical flow based coding method is applied to the video unit is determined based on illuminance information of the video unit, and a bitstream of the video unit is generated based on the determination.

[00113] In some embodiments, whether an optical flow based coding method is applied to the video unit is determined based on illuminance information of the video unit. A bitstream of the video unit is generated based on the determination, and the bitstream is stored in a non-transitory computer-readable recording medium.

[00114] Fig. 28 illustrates a flowchart of a method 2800 for video processing in accordance with some embodiments of the present disclosure. The method 2800 may be implemented during a conversion between a video unit and a bitstream of the video unit.

[00115] At block 2810, during a conversion between a video unit and a bitstream of the video unit, whether an optical flow based coding method is applied to the video unit is determined based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit.

[00116] At block 2820, the conversion is performed based on the determination. In some embodiments, the conversion may comprise encoding the video unit into the bitstream. In some embodiments, the conversion may comprise decoding the video unit from the bitstream.

[00117] According to embodiments of the present disclosure, illuminance information can be considered when determining whether to apply an optical flow based coding method. Compared with the conventional solution, some embodiments of the present disclosure can advantageously improve the coding efficiency and coding performance.

[00118] In some embodiments, whether to enable the optical flow based coding method may be determined based on illuminance information of two reference pictures of the video unit. In some embodiments, if an illuminance change occurs between the two reference pictures, it may be disallowed to enable the optical flow based coding method.

[00119] In some embodiments, a first reference picture in the two reference pictures may be from a first list of reference pictures, and a second reference picture in the two reference pictures may be from a second list of reference pictures. For example, one of the two reference pictures may be from list X, and the other may be from list Y.

[00120] In some embodiments, an absolute picture order coding (POC) distance of the two reference pictures may be equal to twice of an absolute POC distance of one of the two reference pictures relative to the video unit. For example, the current picture of the video unit may be in the middle between the two reference pictures in terms of POC.

[00121] In some embodiments, all of samples or pixels in the two reference pictures may be used for determining the illuminance information. In some embodiment, a part of the samples or pixels in the two reference pictures may be used for determining the illuminance information.

[00122] In some embodiments, whether to enable the optical flow based coding method may be determined based on illuminance information of a current picture and one or more reference pictures associated with the video unit.

[00123] In some embodiments, whether an illuminance change of the video unit occurs may be determined based on the current picture and one or more reference pictures. In some embodiments, whether the illuminance change of the video unit occurs may be determined based on at least one of: an original sample in the one or more reference pictures or a reconstructed sample in the one or more reference pictures.

[00124] In some embodiments, whether the illuminance change of the video unit occurs may be determined based on at least one of: an original sample of the current picture, a reconstructed sample of the current picture, or a prediction sample of the current picture.

[00125] In some embodiments, a set of histograms for the one or more reference pictures may be determined. In this case, in some embodiments, if a difference of the set of histograms is larger than a first threshold value, the illuminance change may occur. In some embodiments, the first threshold value may be set based on at least one of: a size of the current picture, coding information of the video unit, the current picture, or the set of histograms.

[00126] In some embodiments, if an illuminance change occurs among the current picture and the one or more reference pictures, it may not to enable the optical flow based coding method.

[00127] In some embodiments, the illuminance information may comprise a sample value of one or more components between the video unit and the reference video unit of the video unit. In some embodiments, the one or more components comprise a luma component. In some embodiments, the one or more components may comprise one or more chroma components. In some embodiments, the video unit and/or the reference video unit may be a coding block, such as coding unit/prediction unit/transform unit.

[00128] In some embodiments, a first feature of sample value for the video unit may be determined. In some embodiments, a second feature of sample value for the reference video

unit may be determined. In some embodiments, if a difference between the first feature and the second feature is larger than a second threshold value, it may determine not to apply the optical flow based coding method. In some embodiments, the first feature may be determined based on a neighboring sample of the video unit. In some embodiments, the neighboring sample may be adjacent to the video unit. In some embodiments, the neighboring sample may be non-adjacent to the video unit. In some embodiments, a prediction signal for the video unit may be derived. In this case, the first feature may be determined based on the prediction signal. In some embodiments, the prediction signal may be an intra prediction signal. In some embodiments, the second feature may be determined based on a reconstructed sample of the reference video unit.

[00129] In some embodiments, the first feature may comprise at least one of: a mean value of sample value for the video unit, a variance value of sample value for the video unit, or a histogram of sample values for the video unit.

[00130] In some embodiments, the second feature may comprise at least one of: a mean value of sample value for the reference video unit, a variance value of sample value for the reference video unit, or a histogram of sample values for the reference video unit.

[00131] In some embodiments, the second threshold value may be determined based on coding information of the video unit. In some embodiments, the coding information may comprise at least one of: a dimension of the video unit, or a size of the video unit.

[00132] In some embodiments, a determination of an illuminance change of the video unit may be performed in a first level. In some embodiments, a determination of whether to and/or how to use the optical flow based coding method may be performed in a second level. In some embodiments, the first level and the second level may be both block level. In some embodiments, the first level and the second level may be both picture level. In some embodiments, the first level and the second level may be both sub-block level. In some embodiments, the first level may be block level and the second level may be sub-block level. In some embodiments, all of samples or pixels in the first level may be utilized. In some embodiments, a part of the samples or pixels in the first level may be utilized.

[00133] In some embodiments, if the optical flow based coding method is block-level, at least one of detection or a calculation of an illuminance change of the video unit may involve samples in addition to a set of prediction blocks of a current block associated with the video unit. In some embodiments, if the optical flow based coding method is subblock-level, at

least one of detection or a calculation of an illuminance change of the video unit may involve samples in addition to a set of prediction blocks of a current subblock associated with the video unit in a block.

[00134] In some embodiments, the optical flow based coding method may refer to at least one of: a bi-directional optical flow method in which an optical flow is used to refine a bi-prediction signal of a coding block, a prediction refinement with optical flow for affine mode in which the optical flow is used to refine an affine motion compensated prediction, or a coding method in which the optical flow is used to generate or refine a prediction/reconstruction signal of a coding block. In some embodiments, the optical flow based coding method may be a bi-directional optical flow (BDOF). In some embodiments, the optical flow based coding method may be a prediction refinement with optical flow (PROF).

[00135] In some embodiments, the illuminance change of a sample/pixel may refer to the sample/pixel value changes a lot between two different video units (e.g., current picture and its reference picture). In some embodiments, if a change of sample or pixel values between two video units is larger than a first threshold value, an illuminance change may occur. For example, the change may be calculated by $d = \text{abs}(P1 - P2)$ where $P1$ and $P2$ denote two samples/pixels in two different video units and abs represents an absolute value operation. In this case, illuminance change may occur if d is larger than a certain value D . In some embodiments, the first threshold value may be predefined. In some embodiments, the first threshold value may be derived dynamically. For example, the first threshold value may be derived according to decoded information (e.g., adjacent or non-adjacent samples/pixels in current picture/different pictures). In some embodiments, the first threshold value may be indicated in the bitstream.

[00136] In some embodiments, the illuminance change of a video unit may refer to the values of most samples/pixels, and/or the mean value of samples/pixels in the video unit change a lot between two different video units. In some embodiments, if a change of sample or pixel values in the video unit between two video units is larger than a second threshold value, an illuminance change occurs. In some embodiments, if a change of mean values of sample or pixel values in the video unit between two video units is larger than the second threshold value, the illuminance change may occur. For example, the change may be calculated by $d = \text{abs}(m1 - m2)$, where $m1$ and $m2$ denote the output of a function applied to two associated video units and abs represents an absolute value operation. In some embodiments,

the the function may be defined to be a derivation of mean values. For example, two mean values of samples/pixels in two different video units are calculated as m_1 and m_2 , respectively. In some embodiments, the second threshold value may be predefined. In some embodiments, the second threshold value may be derived dynamically. For example, the second threshold value may be derived according to decoded information (e.g., adjacent or non-adjacent samples/pixels in current picture/different pictures). In some embodiments, the second threshold value may be indicated in the bitstream.

[00137] In some embodiments, the illuminance information or an illuminance change of the video unit may comprise at least one of: a luma component, or a chroma component.

[00138] In some embodiments, an indication of whether to apply the optical flow based coding method may be indicated at one of the followings: sequence level, group of pictures level, picture level, slice level, or tile group level. For example, in some embodiments, the indication of whether to apply the optical flow based coding method may be indicated in one of the following: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a dependency parameter set (DPS), a decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter sets (APS), a slice header, or a tile group header.

[00139] In some embodiments, an indication of whether to apply the optical flow based coding method may be included in one of the following: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, a sub-picture, or a region containing more than one sample or pixel.

[00140] In some embodiments, whether the optical flow based coding method is applied may be determined based on coded information of the video unit. The coded information may comprise at least one of: a block size, a colour format, a single and/or dual tree partitioning, a colour component, a slice type, or a picture type.

[00141] In some embodiments, a bitstream of a video may be stored in a non-transitory computer-readable recording medium. The bitstream of the video can be generated by a method performed by a video processing apparatus. According to the method, whether an optical flow based coding method is applied to the video unit is determined based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit, and a bitstream of the video unit is generated based on the determination.

[00142] In some embodiments, whether an optical flow based coding method is applied to the video unit is determined based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit. A bitstream of the video unit is generated based on the determination, and the bitstream is stored in a non-transitory computer-readable recording medium.

[00143] Fig. 29 illustrates a flowchart of a method 2900 for video processing in accordance with some embodiments of the present disclosure. The method 2900 may be implemented during a conversion between a video unit and a bitstream of the video unit.

[00144] At block 2910, during a conversion between a video unit and a bitstream of the video unit, how to apply an optical flow based coding method to the video unit is determined based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit.

[00145] At block 2920, the conversion is performed based on the determination. In some embodiments, the conversion may comprise encoding the video unit into the bitstream. In some embodiments, the conversion may comprise decoding the video unit from the bitstream.

[00146] According to embodiments of the present disclosure, illuminance information can be considered when determining how to apply an optical flow based coding method. Compared with the conventional solution, some embodiments of the present disclosure can advantageously improve the coding efficiency and coding performance.

[00147] In some embodiments, the optical flow based coding process may be applied to the video unit based on whether an illuminance change of the video unit occurs. In some embodiments, the illuminance change may be included in a process of the optical flow based coding method.

[00148] In some embodiments, a value may be subtracted in calculation of a gradient in the process of the optical flow based coding method.

[00149] In some embodiments, during the process of the optical flow based coding method, a first value may be subtracted from a first sample or pixel in a first prediction block of the video unit. In some embodiments, a second value may be subtracted from a second sample or pixel in a second prediction block of the video unit. In some embodiments, a difference of the first sample or pixel and the second sample or pixel may be determined.

[00150] In some embodiments, a prediction block of the video unit may be revised via a function used in the optical flow based coding method. In some embodiments, the function may be a linear function: $f(P_i)=a*P_i+b$, where a and b represent linear parameters, respectively, and P_i represents a sample or pixel in the prediction block of the video unit. Instead of using the first/second prediction blocks directly obtained from motion information, it is proposed to firstly revise the obtained prediction blocks via a function, i.e. $f(P_i)$ is used in the optical flow procedure, instead of the sample value P_i .

[00151] In some embodiments, at least one of the linear parameters may be determined based on one of: a coding tool (for example, LIC and/or BCW), or a set of neighbor samples or pixels of the video unit. In some embodiments, at least one of the linear parameters may be indicated in the bitstream. In some embodiments, the function may be different for different prediction blocks of the video unit.

[00152] In some embodiments, the function may be a non-linear function. In some embodiments, a set of model parameters of the illuminance change may be jointly optimized with a set of parameters of the optical flow based coding method. In some embodiments, the set of model parameters of the illuminance change and the set of parameters of the optical flow based coding method may be updated with least square regression method iteratively.

[00153] In some embodiments, a determination of an illuminance change of the video unit may be performed in a first level, and a determination of how to the optical flow based coding method may be performed in a second level.

[00154] In some embodiments, the first level and the second level may be both block level. In some embodiments, the first level and the second level may be both picture level. In some embodiments, the first level and the second level may be both sub-block level. In some embodiments, the first level may be block level and the second level may be sub-block level. In some embodiments, all of samples or pixels in the first level may be utilized. In some embodiments, a part of the samples or pixels in the first level may be utilized.

[00155] In some embodiments, an indication of how to apply the optical flow based coding method may be indicated at one of the followings: sequence level, group of pictures level, picture level, slice level, or tile group level. For example, in some embodiments, the indication of how to apply the optical flow based coding method may be indicated in one of the following: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a dependency parameter set (DPS), a decoding capability information

(DCI), a picture parameter set (PPS), an adaptation parameter sets (APS), a slice header, or a tile group header.

[00156] In some embodiments, an indication of how to apply the optical flow based coding method may be included in one of the following: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, a sub-picture, or a region containing more than one sample or pixel.

[00157] In some embodiments, how the optical flow based coding method is applied may be determined based on based on coded information of the video unit. The coded information may comprise at least one of: a block size, a colour format, a single and/or dual tree partitioning, a colour component, a slice type, or a picture type.

[00158] In some embodiments, a bitstream of a video may be stored in a non-transitory computer-readable recording medium. The bitstream of the video can be generated by a method performed by a video processing apparatus. According to the method, how to apply an optical flow based coding method to the video unit is determined based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit, and a bitstream of the video unit is generated based on the determination.

[00159] In some embodiments, how to apply an optical flow based coding method to the video unit is determined based on illuminance information associated with at least one of: the video unit or a reference video unit of the video unit. A bitstream of the video unit is generated based on the determination, and the bitstream is stored in a non-transitory computer-readable recording medium.

[00160] Implementations of the present disclosure can be described in view of the following clauses, the features of which can be combined in any reasonable manner.

[00161] Clause 1. A method of video processing, comprising: determining, during a conversion between a video unit and a bitstream of the video unit, information on applying an optical flow based coding method to the video unit based on illuminance information of the video unit; and performing the conversion based on the information.

[00162] Clause 2. The method of clause 1, wherein the information on applying the optical flow based coding method comprises at least one of: whether to apply the optical flow based

coding method to the video unit, or how to apply optical flow based coding method to the video unit.

[00163] Clause 3. The method of clause 1, wherein determining the information on applying the optical flow based coding method to the video unit comprises: determining whether an illuminance change of the video unit occurs; and in response to that the illuminance change occurs, determining that the optical flow based coding method is not applied to the video unit.

[00164] Clause 4. The method of clause 3, wherein a syntax element in the bitstream indicates whether the illuminance change occurs.

[00165] Clause 5. The method of clause 3, wherein determining whether the illuminance change of the video unit occurs comprises: determining whether the illuminance change of the video unit occurs based on at least one of: whether a coding tool is applied to the video unit, or how the coding tool is applied to the video unit.

[00166] Clause 6. The method of clause 3, wherein determining whether the illuminance change of the video unit occurs comprises: determining whether the illuminance change of the video unit occurs based on a set of neighbor samples of the video unit.

[00167] Clause 7. The method of clause 3, further comprising: in response to that the illuminance change occurs on a sample or a pixel of the video unit, determining that the optical flow based coding method is not applied to the sample or the pixel of the video unit.

[00168] Clause 8. The method of clause 1, further comprising: applying the optical flow based coding process to the video unit based on whether an illuminance change of the video unit occurs.

[00169] Clause 9. The method of clause 8, wherein the illuminance change is included in a process of the optical flow based coding method.

[00170] Clause 10. The method of clause 9, wherein a value is subtracted in calculation of a gradient in the process of the optical flow based coding method.

[00171] Clause 11. The method of clause 9, further comprising: during the process of the optical flow based coding method, subtracting a first value from a first sample or pixel in a first prediction block of the video unit; subtracting a second value from a second sample or pixel in a second prediction block of the video unit; determining a difference of the first sample or pixel and the second sample or pixel.

[00172] Clause 12. The method of clause 9, wherein a prediction block of the video unit is revised via a function used in the optical flow based coding method.

[00173] Clause 13. The method of clause 12, wherein the function is a linear function: $f(P_i)=a*P_i+b$, where a and b represent linear parameters, respectively, and P_i represents a sample or pixel in the prediction block of the video unit.

[00174] Clause 14. The method of clause 13, wherein at least one of the linear parameters is determined based on one of: a coding tool, or a set of neighbor samples or pixels of the video unit.

[00175] Clause 15. The method of clause 12, wherein at least one of the linear parameters is indicated in the bitstream.

[00176] Clause 16. The method of clause 12, wherein the function is different for different prediction blocks of the video unit.

[00177] Clause 17. The method of clause 12, wherein the function is a non-linear function.

[00178] Clause 18. The method of clause 9, wherein a set of model parameters of the illuminance change are jointly optimized with a set of parameters of the optical flow based coding method.

[00179] Clause 19. The method of clause 18, wherein the set of model parameters of the illuminance change and the set of parameters of the optical flow based coding method are updated with least square regression method iteratively.

[00180] Clause 20. The method of clause 1, wherein a determination of an illuminance change of the video unit is performed in a first level, and a determination of how to and/or whether to apply the optical flow based coding method is performed in a second level.

[00181] Clause 21. The method of clause 20, wherein the first level and the second level are both block level, or wherein the first level and the second level are both picture level, or wherein the first level and the second level are both sub-block level, or wherein the first level is block level and the second level is sub-block level.

[00182] Clause 22. The method of clause 20, wherein all of samples or pixels in the first level are utilized, or wherein a part of the samples or pixels in the first level is utilized.

[00183] Clause 23. The method of clause 1, wherein if the optical flow based coding method is block-level, at least one of detection or a calculation of an illuminance change of the video unit involves samples in addition to a set of prediction blocks of a current block associated with the video unit.

[00184] Clause 24. The method of clause 1, wherein if the optical flow based coding method is subblock-level, at least one of detection or a calculation of an illuminance change of the video unit involves samples in addition to a set of prediction blocks of a current sub-block associated with the video unit in a block. [bullet 5]

[00185] Clause 25. The method of clause 1, wherein how to apply the optical flow based coding method is determined based on at least one of: whether a coding tool is applied to the video unit, or how the coding tool is applied to the video unit.

[00186] Clause 26. The method of clause 1, wherein whether the optical flow based coding method is applied to the video unit is determined based on at least one of: whether a coding tool is applied to the video unit, or how the coding tool is applied to the video unit.

[00187] Clause 27. The method of clause 25 or 26, wherein the coding tool comprises at least one of: a location illumination compensation method, a bi-prediction with coding unit level weight method, or an affine compensation method.

[00188] Clause 28. The method of clause 26, wherein if the coding tool is applied to the video unit, the optical flow based coding method is not applied to the video unit.

[00189] Clause 29. The method of clause 26, wherein if the coding tool is applied to the video unit, the optical flow based coding method is applied to the video unit.

[00190] Clause 30. The method of clause 1, wherein the optical flow based coding method comprises at least one of: a bi-directional optical flow method in which an optical flow is used to refine a bi-prediction signal of a coding block, a prediction refinement with optical flow for affine mode in which the optical flow is used to refine an affine motion compensated prediction, or a coding method in which the optical flow is used to generate or refine a prediction/reconstruction signal of a coding block.

[00191] Clause 31. The method of clause 30, wherein the optical flow based coding method is a bi-directional optical flow (BDOF), or wherein the optical flow based coding method is a prediction refinement with optical flow (PROF).

[00192] Clause 32. The method of clause 1, wherein if a change of sample or pixel values between two video units is larger than a first threshold value, an illuminance change occurs.

[00193] Clause 33. The method of clause 32, wherein the change is calculated by: $d = \text{abs}(P1 - P2)$, wherein P1 and P2 represent two samples or pixels in the two video units, respectively, and abs represents an absolute value operation.

[00194] Clause 34. The method of clause 32, wherein the first threshold value is predefined, or wherein the first threshold value is determined dynamically, or wherein the first threshold value is indicated in the bitstream.

[00195] Clause 35. The method of clause 1, wherein if a change of sample or pixel values in the video unit between two video units is larger than a second threshold value, an illuminance change occurs, or wherein if a change of mean values of sample or pixel values in the video unit between two video units is larger than the second threshold value, the illuminance change occurs.

[00196] Clause 36. The method of clause 35, wherein the change is calculated by: $d = \text{abs}(m1 - m2)$, wherein m1 and m2 represent output of a function applied to the two associated video units, respectively, and abs represents an absolute value operation.

[00197] Clause 37. The method of clause 36, wherein the function is defined to be a derivation of mean values.

[00198] Clause 38. The method of clause 35, wherein the second threshold value is predefined, or wherein the second threshold value is determined dynamically, or wherein the second threshold value is indicated in the bitstream.

[00199] Clause 39. The method of clause 1, wherein the video unit comprises one of: a color component, a sub-picture, a slice, a tile, a coding tree unit (CTU), a CTU row, a group of CTUs, a coding unit (CU), a prediction unit (PU), a transform unit (TU), a coding tree block (CTB), a coding block (CB), a prediction block (PB), a transform block (TB), a block, a sub-block of a block, a sub-region within the block, or a region that comprises more than one sample or pixel.

[00200] Clause 40. The method of clause 1, wherein the illuminance information or an illuminance change of the video unit comprises at least one of: a luma component, or a chroma component.

[00201] Clause 41. The method of clause 1, wherein the conversion includes encoding the video unit into the bitstream.

[00202] Clause 42. The method of clause 1, wherein the conversion includes decoding the video unit from the bitstream.

[00203] Clause 43. The method of any of clauses 1-42, wherein an indication of whether to and/or how to apply the optical flow based coding method is indicated at one of the followings: sequence level, group of pictures level, picture level, slice level, or tile group level.

[00204] Clause 44. The method of any of clauses 1-42, wherein an indication of whether to and/or how to apply the optical flow based coding method is indicated in one of the following: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a dependency parameter set (DPS), a decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter sets (APS), a slice header, or a tile group header.

[00205] Clause 45. The method of any of clauses 1-42, wherein an indication of whether to and/or how to apply the optical flow based coding method is included in one of the following: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, a sub-picture, or a region containing more than one sample or pixel.

[00206] Clause 46. The method of any of clauses 1-42, further comprising: determining, based on coded information of the video unit, whether and/or how the optical flow based coding method is applied, the coded information including at least one of: a block size, a colour format, a single and/or dual tree partitioning, a colour component, a slice type, or a picture type.

[00207] Clause 47. An apparatus for processing video data comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to perform a method in accordance with any of Clauses 1-46.

[00208] Clause 48. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of Clauses 1-46.

[00209] Clause 49. A non-transitory computer-readable recording medium storing a bitstream of a video which is generated by a method performed by a video processing apparatus, wherein the method comprises: A non-transitory computer-readable recording medium storing a bitstream of a video which is generated by a method performed by a video processing apparatus, wherein the method comprises: determining information on applying an optical flow based coding method to a video unit based on illuminance information of the video unit; and generating a bitstream of the video unit based on the information.

[00210] Clause 50. A method for storing a bitstream of a video, comprising: determining information on applying an optical flow based coding method to a video unit based on illuminance information of the video unit; generating a bitstream of the video unit based on the information; and storing the bitstream in a non-transitory computer-readable recording medium.

Example Device

[00211] Fig. 30 illustrates a block diagram of a computing device 3000 in which various embodiments of the present disclosure can be implemented. The computing device 3000 may be implemented as or included in the source device 110 (or the video encoder 114 or 200) or the destination device 120 (or the video decoder 124 or 300).

[00212] It would be appreciated that the computing device 3000 shown in Fig. 30 is merely for purpose of illustration, without suggesting any limitation to the functions and scopes of the embodiments of the present disclosure in any manner.

[00213] As shown in Fig. 30, the computing device 3000 includes a general-purpose computing device 3000. The computing device 3000 may at least comprise one or more processors or processing units 3010, a memory 3020, a storage unit 3030, one or more communication units 3040, one or more input devices 3050, and one or more output devices 3060.

[00214] In some embodiments, the computing device 3000 may be implemented as any user terminal or server terminal having the computing capability. The server terminal may be a server, a large-scale computing device or the like that is provided by a service provider. The user terminal may for example be any type of mobile terminal, fixed terminal, or portable terminal, including a mobile phone, station, unit, device, multimedia computer, multimedia tablet, Internet node, communicator, desktop computer, laptop computer, notebook computer, netbook computer, tablet computer, personal communication system (PCS) device, personal

navigation device, personal digital assistant (PDA), audio/video player, digital camera/video camera, positioning device, television receiver, radio broadcast receiver, E-book device, gaming device, or any combination thereof, including the accessories and peripherals of these devices, or any combination thereof. It would be contemplated that the computing device 3000 can support any type of interface to a user (such as “wearable” circuitry and the like).

[00215] The processing unit 3010 may be a physical or virtual processor and can implement various processes based on programs stored in the memory 3020. In a multi-processor system, multiple processing units execute computer executable instructions in parallel so as to improve the parallel processing capability of the computing device 3000. The processing unit 3010 may also be referred to as a central processing unit (CPU), a microprocessor, a controller or a microcontroller.

[00216] The computing device 3000 typically includes various computer storage medium. Such medium can be any medium accessible by the computing device 3000, including, but not limited to, volatile and non-volatile medium, or detachable and non-detachable medium. The memory 3020 can be a volatile memory (for example, a register, cache, Random Access Memory (RAM)), a non-volatile memory (such as a Read-Only Memory (ROM), Electrically Erasable Programmable Read-Only Memory (EEPROM), or a flash memory), or any combination thereof. The storage unit 3030 may be any detachable or non-detachable medium and may include a machine-readable medium such as a memory, flash memory drive, magnetic disk or another other media, which can be used for storing information and/or data and can be accessed in the computing device 3000.

[00217] The computing device 3000 may further include additional detachable/non-detachable, volatile/non-volatile memory medium. Although not shown in Fig. 30, it is possible to provide a magnetic disk drive for reading from and/or writing into a detachable and non-volatile magnetic disk and an optical disk drive for reading from and/or writing into a detachable non-volatile optical disk. In such cases, each drive may be connected to a bus (not shown) via one or more data medium interfaces.

[00218] The communication unit 3040 communicates with a further computing device via the communication medium. In addition, the functions of the components in the computing device 3000 can be implemented by a single computing cluster or multiple computing machines that can communicate via communication connections. Therefore, the computing

device 3000 can operate in a networked environment using a logical connection with one or more other servers, networked personal computers (PCs) or further general network nodes.

[00219] The input device 3050 may be one or more of a variety of input devices, such as a mouse, keyboard, tracking ball, voice-input device, and the like. The output device 3060 may be one or more of a variety of output devices, such as a display, loudspeaker, printer, and the like. By means of the communication unit 3040, the computing device 3000 can further communicate with one or more external devices (not shown) such as the storage devices and display device, with one or more devices enabling the user to interact with the computing device 3000, or any devices (such as a network card, a modem and the like) enabling the computing device 3000 to communicate with one or more other computing devices, if required. Such communication can be performed via input/output (I/O) interfaces (not shown).

[00220] In some embodiments, instead of being integrated in a single device, some or all components of the computing device 3000 may also be arranged in cloud computing architecture. In the cloud computing architecture, the components may be provided remotely and work together to implement the functionalities described in the present disclosure. In some embodiments, cloud computing provides computing, software, data access and storage service, which will not require end users to be aware of the physical locations or configurations of the systems or hardware providing these services. In various embodiments, the cloud computing provides the services via a wide area network (such as Internet) using suitable protocols. For example, a cloud computing provider provides applications over the wide area network, which can be accessed through a web browser or any other computing components. The software or components of the cloud computing architecture and corresponding data may be stored on a server at a remote position. The computing resources in the cloud computing environment may be merged or distributed at locations in a remote data center. Cloud computing infrastructures may provide the services through a shared data center, though they behave as a single access point for the users. Therefore, the cloud computing architectures may be used to provide the components and functionalities described herein from a service provider at a remote location. Alternatively, they may be provided from a conventional server or installed directly or otherwise on a client device.

[00221] The computing device 3000 may be used to implement video encoding/decoding in embodiments of the present disclosure. The memory 3020 may include one or more video

coding modules 3025 having one or more program instructions. These modules are accessible and executable by the processing unit 3010 to perform the functionalities of the various embodiments described herein.

[00222] In the example embodiments of performing video encoding, the input device 3050 may receive video data as an input 3070 to be encoded. The video data may be processed, for example, by the video coding module 3025, to generate an encoded bitstream. The encoded bitstream may be provided via the output device 3060 as an output 3080.

[00223] In the example embodiments of performing video decoding, the input device 3050 may receive an encoded bitstream as the input 3070. The encoded bitstream may be processed, for example, by the video coding module 3025, to generate decoded video data. The decoded video data may be provided via the output device 3060 as the output 3080.

[00224] While this disclosure has been particularly shown and described with references to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the present application as defined by the appended claims. Such variations are intended to be covered by the scope of this present application. As such, the foregoing description of embodiments of the present application is not intended to be limiting.

I/We Claim:

1. A method of video processing, comprising:
determining, during a conversion between a video unit and a bitstream of the video unit, information on applying an optical flow based coding method to the video unit based on illuminance information of the video unit; and
performing the conversion based on the information.
2. The method of claim 1, wherein the information on applying the optical flow based coding method comprises at least one of:
whether to apply the optical flow based coding method to the video unit, or
how to apply optical flow based coding method to the video unit.
3. The method of claim 1, wherein determining the information on applying the optical flow based coding method to the video unit comprises:
determining whether an illuminance change of the video unit occurs; and
in response to that the illuminance change occurs, determining that the optical flow based coding method is not applied to the video unit.
4. The method of claim 3, wherein a syntax element in the bitstream indicates whether the illuminance change occurs.
5. The method of claim 3, wherein determining whether the illuminance change of the video unit occurs comprises:
determining whether the illuminance change of the video unit occurs based on at least one of:
whether a coding tool is applied to the video unit, or
how the coding tool is applied to the video unit.

6. The method of claim 3, wherein determining whether the illuminance change of the video unit occurs comprises:

determining whether the illuminance change of the video unit occurs based on a set of neighbor samples of the video unit.

7. The method of claim 3, further comprising:

in response to that the illuminance change occurs on a sample or a pixel of the video unit, determining that the optical flow based coding method is not applied to the sample or the pixel of the video unit.

8. The method of claim 1, further comprising:

applying the optical flow based coding process to the video unit based on whether an illuminance change of the video unit occurs.

9. The method of claim 8, wherein the illuminance change is included in a process of the optical flow based coding method.

10. The method of claim 9, wherein a value is subtracted in calculation of a gradient in the process of the optical flow based coding method.

11. The method of claim 9, further comprising:

during the process of the optical flow based coding method,
subtracting a first value from a first sample or pixel in a first prediction block of the video unit;

subtracting a second value from a second sample or pixel in a second prediction block of the video unit;

determining a difference of the first sample or pixel and the second sample or pixel.

12. The method of claim 9, wherein a prediction block of the video unit is revised via a function used in the optical flow based coding method.

13. The method of claim 12, wherein the function is a linear function:

$$f(P_i) = a * P_i + b,$$

where a and b represent linear parameters, respectively, and P_i represents a sample or pixel in the prediction block of the video unit.

14. The method of claim 13, wherein at least one of the linear parameters is determined based on one of:

a coding tool, or

a set of neighbor samples or pixels of the video unit.

15. The method of claim 12, wherein at least one of the linear parameters is indicated in the bitstream.

16. The method of claim 12, wherein the function is different for different prediction blocks of the video unit.

17. The method of claim 12, wherein the function is a non-linear function.

18. The method of claim 9, wherein a set of model parameters of the illuminance change are jointly optimized with a set of parameters of the optical flow based coding method.

19. The method of claim 18, wherein the set of model parameters of the illuminance change and the set of parameters of the optical flow based coding method are updated with least square regression method iteratively.

20. The method of claim 1, wherein a determination of an illuminance change of the video unit is performed in a first level, and

a determination of how to and/or whether to apply the optical flow based coding method is performed in a second level.

21. The method of claim 20, wherein the first level and the second level are both block level, or

wherein the first level and the second level are both picture level, or

wherein the first level and the second level are both sub-block level, or

wherein the first level is block level and the second level is sub-block level.

22. The method of claim 20, wherein all of samples or pixels in the first level are utilized, or

wherein a part of the samples or pixels in the first level is utilized.

23. The method of claim 1, wherein if the optical flow based coding method is block-level, at least one of detection or a calculation of an illuminance change of the video unit involves samples in addition to a set of prediction blocks of a current block associated with the video unit.

24. The method of claim 1, wherein if the optical flow based coding method is sub-block-level, at least one of detection or a calculation of an illuminance change of the video unit involves samples in addition to a set of prediction blocks of a current subblock associated with the video unit in a block.

25. The method of claim 1, wherein how to apply the optical flow based coding method is determined based on at least one of:

whether a coding tool is applied to the video unit, or

how the coding tool is applied to the video unit.

26. The method of claim 1, wherein whether the optical flow based coding method is applied to the video unit is determined based on at least one of:

whether a coding tool is applied to the video unit, or

how the coding tool is applied to the video unit.

27. The method of claim 25 or 26, wherein the coding tool comprises at least one of:

a location illumination compensation method,

a bi-prediction with coding unit level weight method, or

an affine compensation method.

28. The method of claim 26, wherein if the coding tool is applied to the video unit, the optical flow based coding method is not applied to the video unit.

29. The method of claim 26, wherein if the coding tool is applied to the video unit, the optical flow based coding method is applied to the video unit.

30. The method of claim 1, wherein the optical flow based coding method comprises at least one of:

a bi-directional optical flow method in which an optical flow is used to refine a bi-prediction signal of a coding block,

a prediction refinement with optical flow for affine mode in which the optical flow is used to refine an affine motion compensated prediction, or

a coding method in which the optical flow is used to generate or refine a prediction/reconstruction signal of a coding block.

31. The method of claim 30, wherein the optical flow based coding method is a bi-directional optical flow (BDOF), or

wherein the optical flow based coding method is a prediction refinement with optical flow (PROF).

32. The method of claim 1, wherein if a change of sample or pixel values between two video units is larger than a first threshold value, an illuminance change occurs.

33. The method of claim 32, wherein the change is calculated by:

$$d = \text{abs}(P1 - P2),$$

wherein P1 and P2 represent two samples or pixels in the two video units, respectively, and abs represents an absolute value operation.

34. The method of claim 32, wherein the first threshold value is predefined, or

wherein the first threshold value is determined dynamically, or

wherein the first threshold value is indicated in the bitstream.

35. The method of claim 1, wherein if a change of sample or pixel values in the video unit between two video units is larger than a second threshold value, an illuminance change occurs, or

wherein if a change of mean values of sample or pixel values in the video unit between two video units is larger than the second threshold value, the illuminance change occurs.

36. The method of claim 35, wherein the change is calculated by:

$$d = \text{abs}(m1 - m2),$$

wherein m1 and m2 represent output of a function applied to the two associated video units, respectively, and abs represents an absolute value operation.

37. The method of claim 36, wherein the function is defined to be a derivation of mean values.

38. The method of claim 35, wherein the second threshold value is predefined, or wherein the second threshold value is determined dynamically, or wherein the second threshold value is indicated in the bitstream.

39. The method of claim 1, wherein the video unit comprises one of:

a color component,

a sub-picture,

a slice,

a tile,

a coding tree unit (CTU),

a CTU row,

a group of CTUs,

a coding unit (CU),

a prediction unit (PU),

a transform unit (TU),

a coding tree block (CTB),

a coding block (CB),

a prediction block (PB),

a transform block (TB),

a block,

a sub-block of a block,

a sub-region within the block, or

a region that comprises more than one sample or pixel.

40. The method of claim 1, wherein the illuminance information or an illuminance change of the video unit comprises at least one of:

- a luma component, or
- a chroma component.

41. The method of claim 1, wherein the conversion includes encoding the video unit into the bitstream.

42. The method of claim 1, wherein the conversion includes decoding the video unit from the bitstream.

43. The method of any of claims 1-42, wherein an indication of whether to and/or how to apply the optical flow based coding method is indicated at one of the followings:

- sequence level,
- group of pictures level,
- picture level,
- slice level, or
- tile group level.

44. The method of any of claims 1-42, wherein an indication of whether to and/or how to apply the optical flow based coding method is indicated in one of the following:

- a sequence header,
- a picture header,
- a sequence parameter set (SPS),
- a video parameter set (VPS),
- a dependency parameter set (DPS),
- a decoding capability information (DCI),
- a picture parameter set (PPS),

an adaptation parameter sets (APS),
a slice header, or
a tile group header.

45. The method of any of claims 1-42, wherein an indication of whether to and/or how to apply the optical flow based coding method is included in one of the following:

a prediction block (PB),
a transform block (TB),
a coding block (CB),
a prediction unit (PU),
a transform unit (TU),
a coding unit (CU),
a virtual pipeline data unit (VPDU),
a coding tree unit (CTU),
a CTU row,
a slice,
a tile,
a sub-picture, or
a region containing more than one sample or pixel.

46. The method of any of claims 1-42, further comprising:
determining, based on coded information of the video unit, whether and/or how the optical flow based coding method is applied, the coded information including at least one of:

a block size,
a colour format,
a single and/or dual tree partitioning,
a colour component,
a slice type, or

a picture type.

47. An apparatus for processing video data comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to perform a method in accordance with any of claims 1-46.

48. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of claims 1-46.

49. A non-transitory computer-readable recording medium storing a bitstream of a video which is generated by a method performed by a video processing apparatus, wherein the method comprises:

determining information on applying an optical flow based coding method to a video unit based on illuminance information of the video unit; and

generating a bitstream of the video unit based on the information.

50. A method for storing bitstream of a video, comprising:

determining information on applying an optical flow based coding method to a video unit based on illuminance information of the video unit;

generating a bitstream of the video unit based on the information; and

storing the bitstream in a non-transitory computer-readable recording medium.

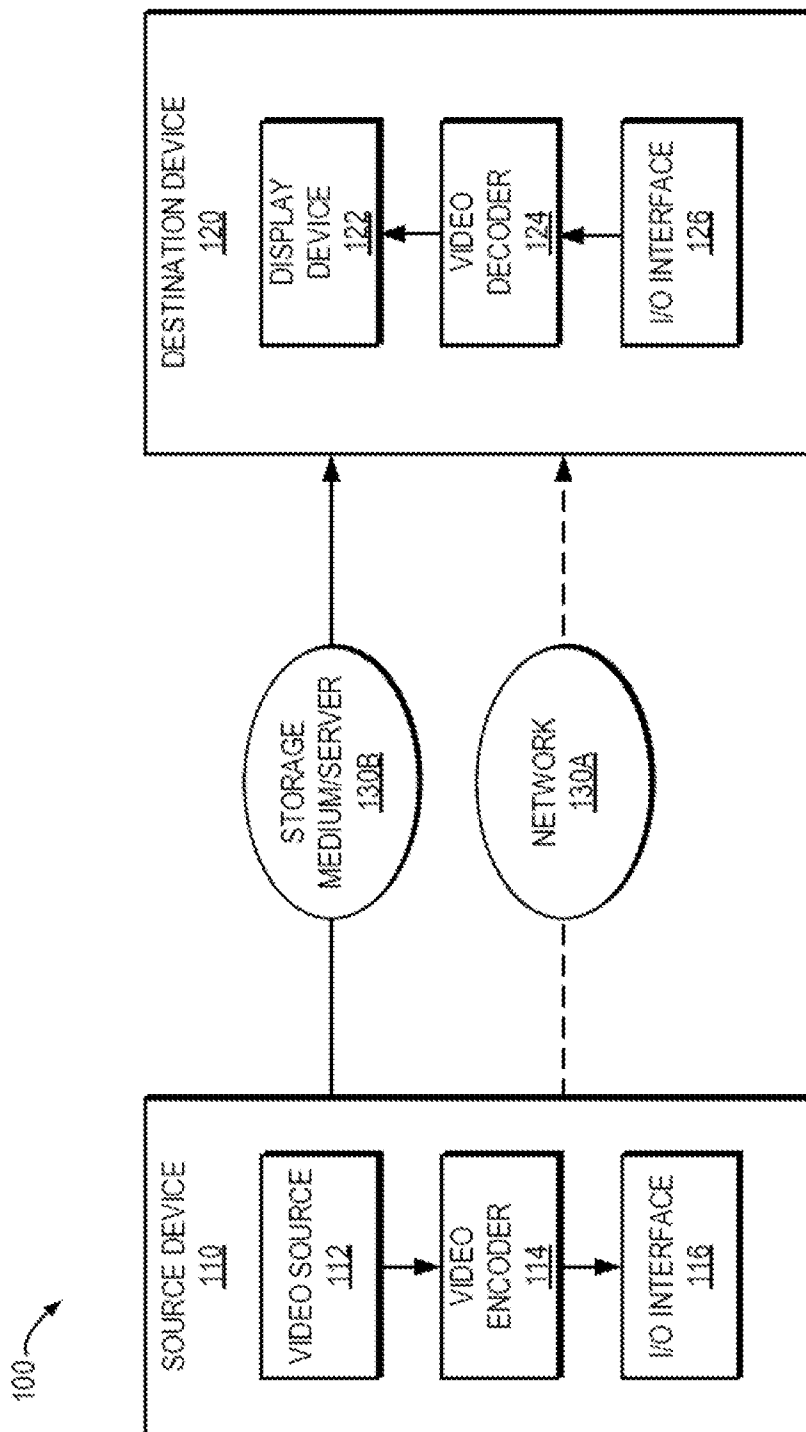


Fig. 1

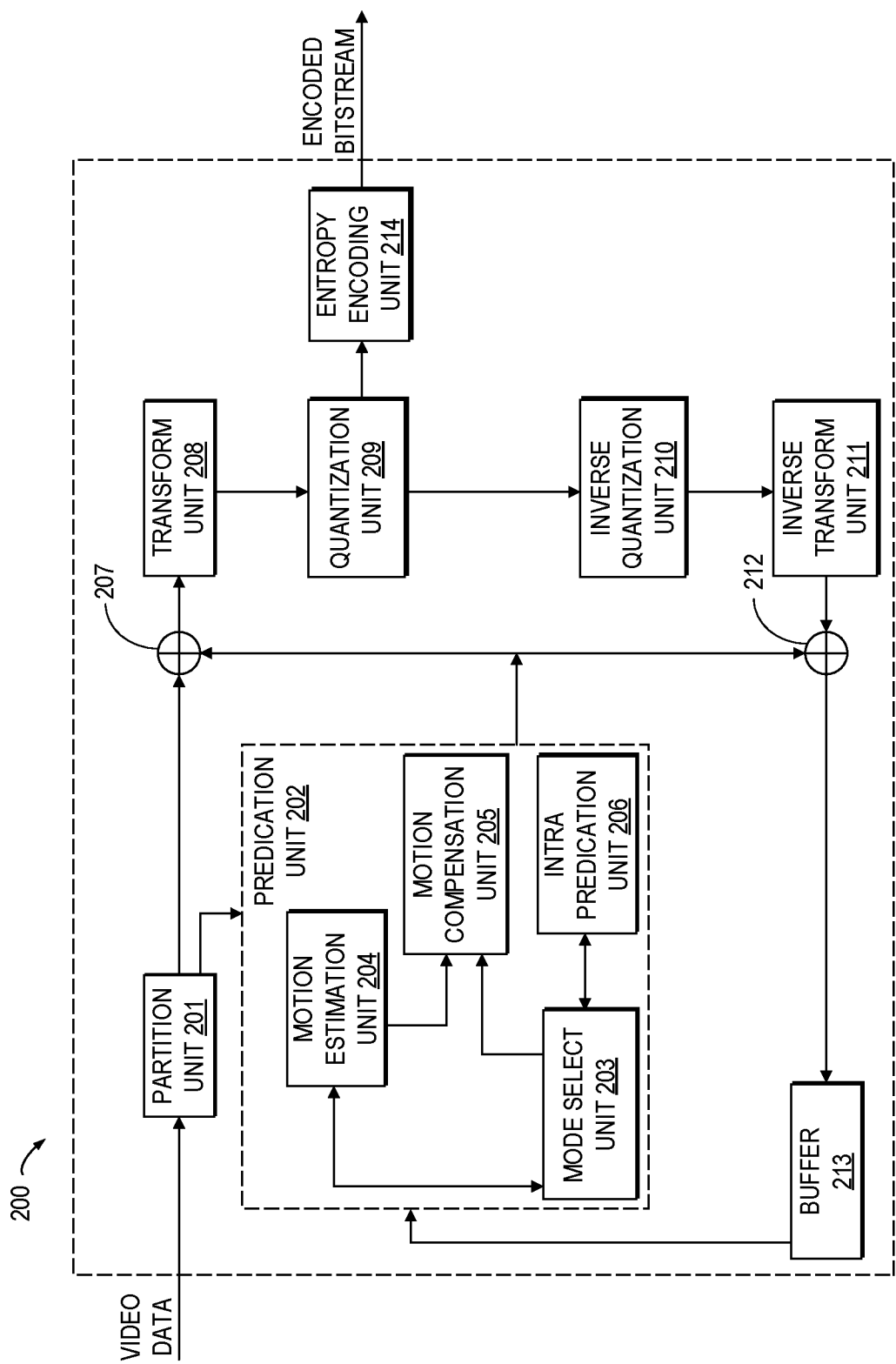


Fig. 2

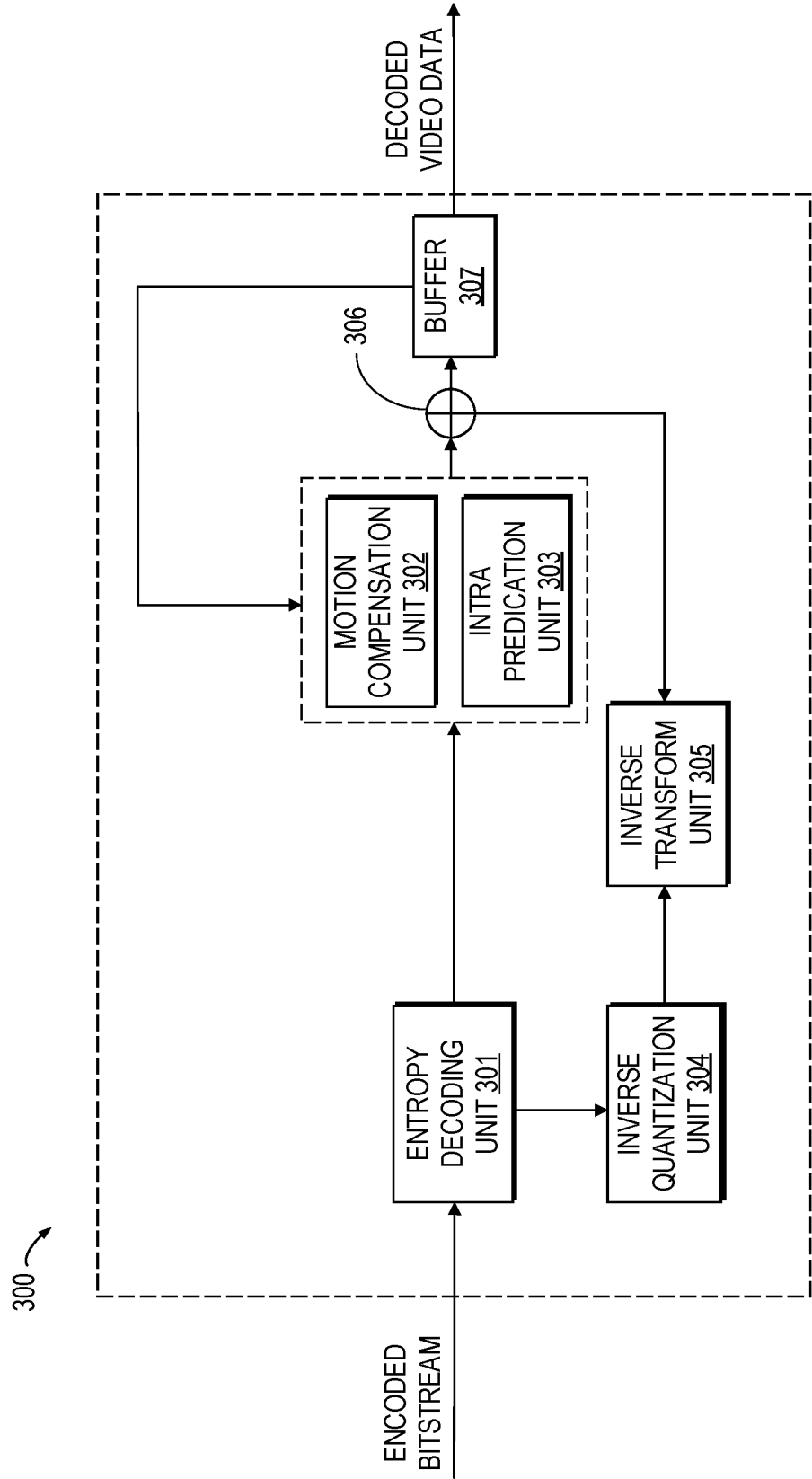
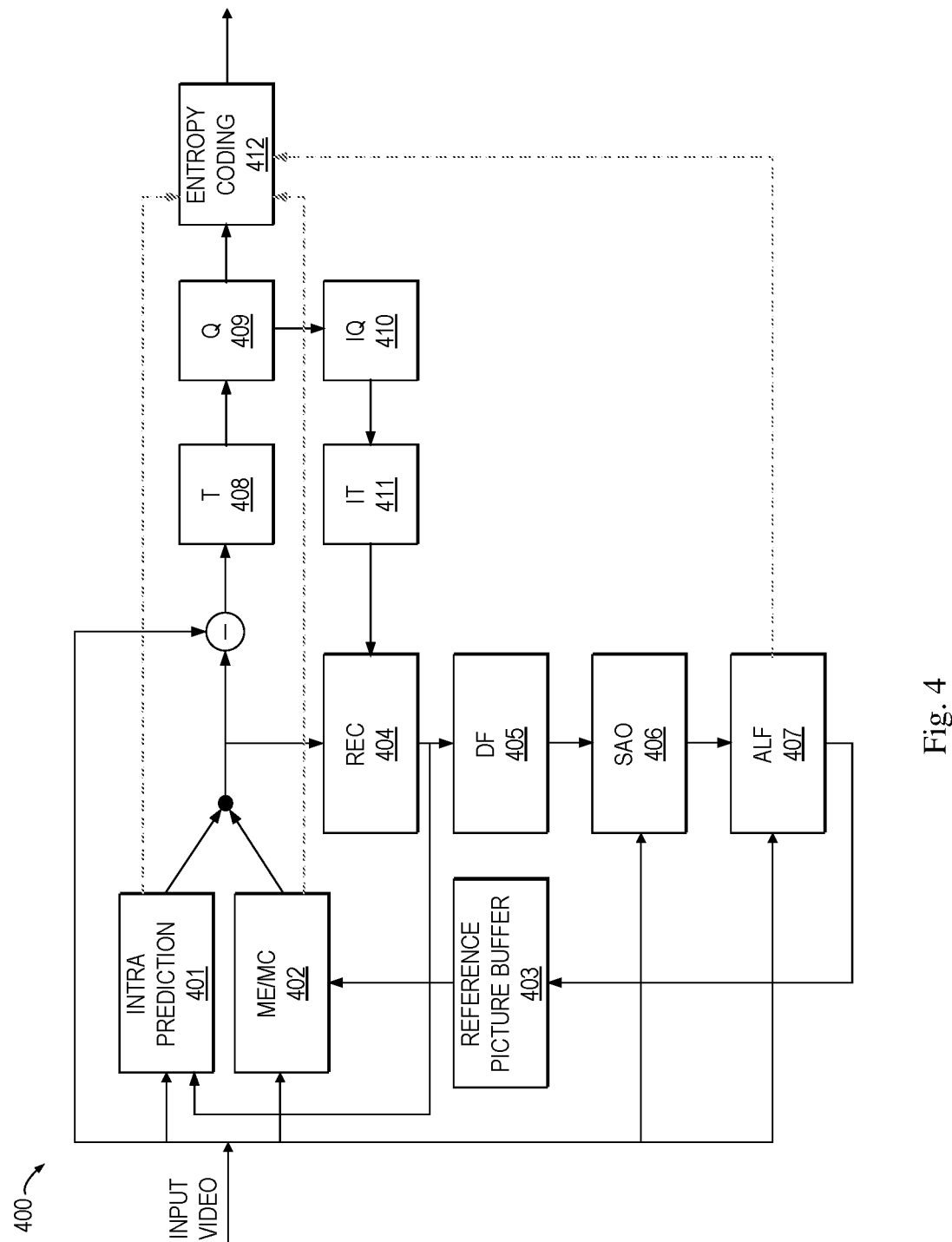


Fig. 3



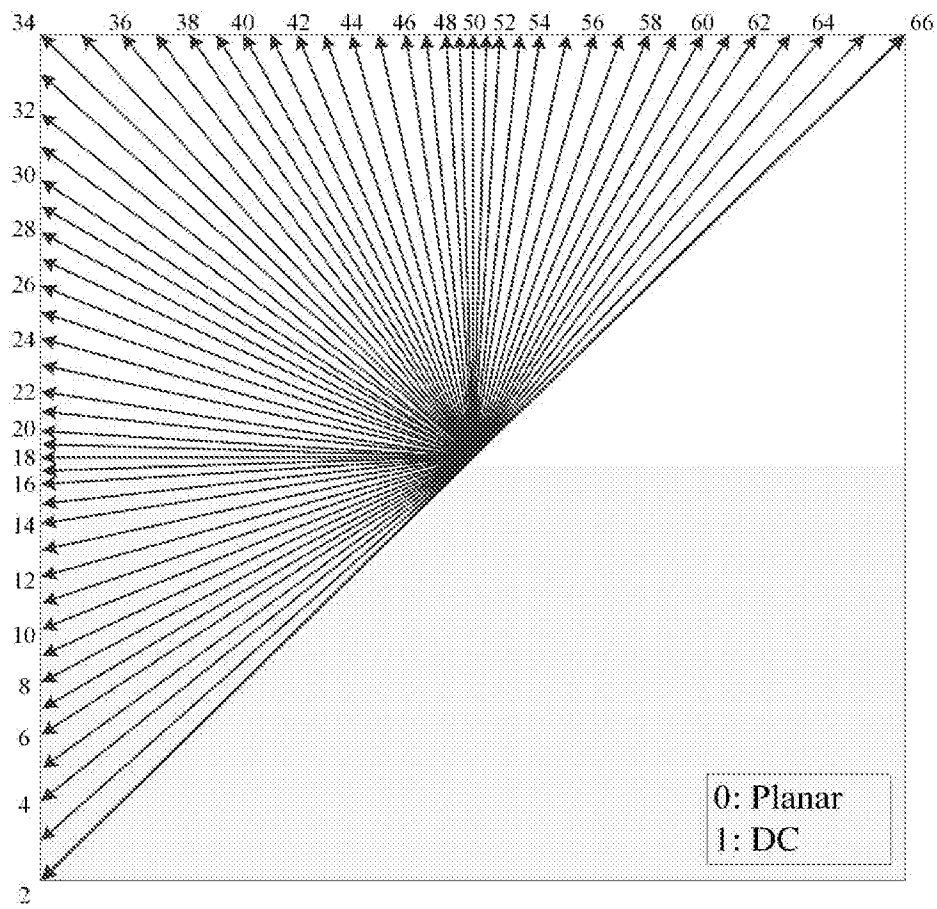


Fig. 5

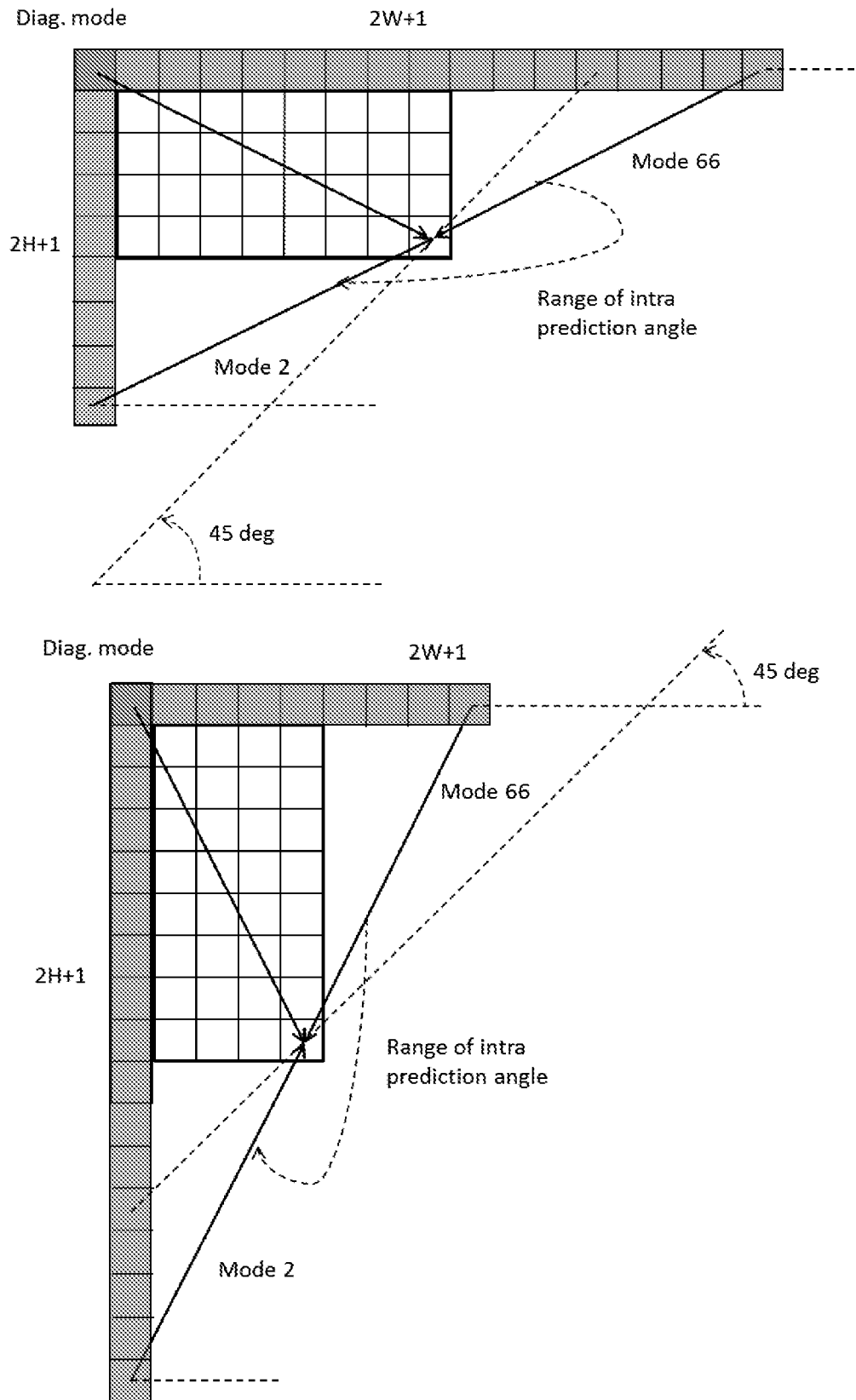


Fig. 6

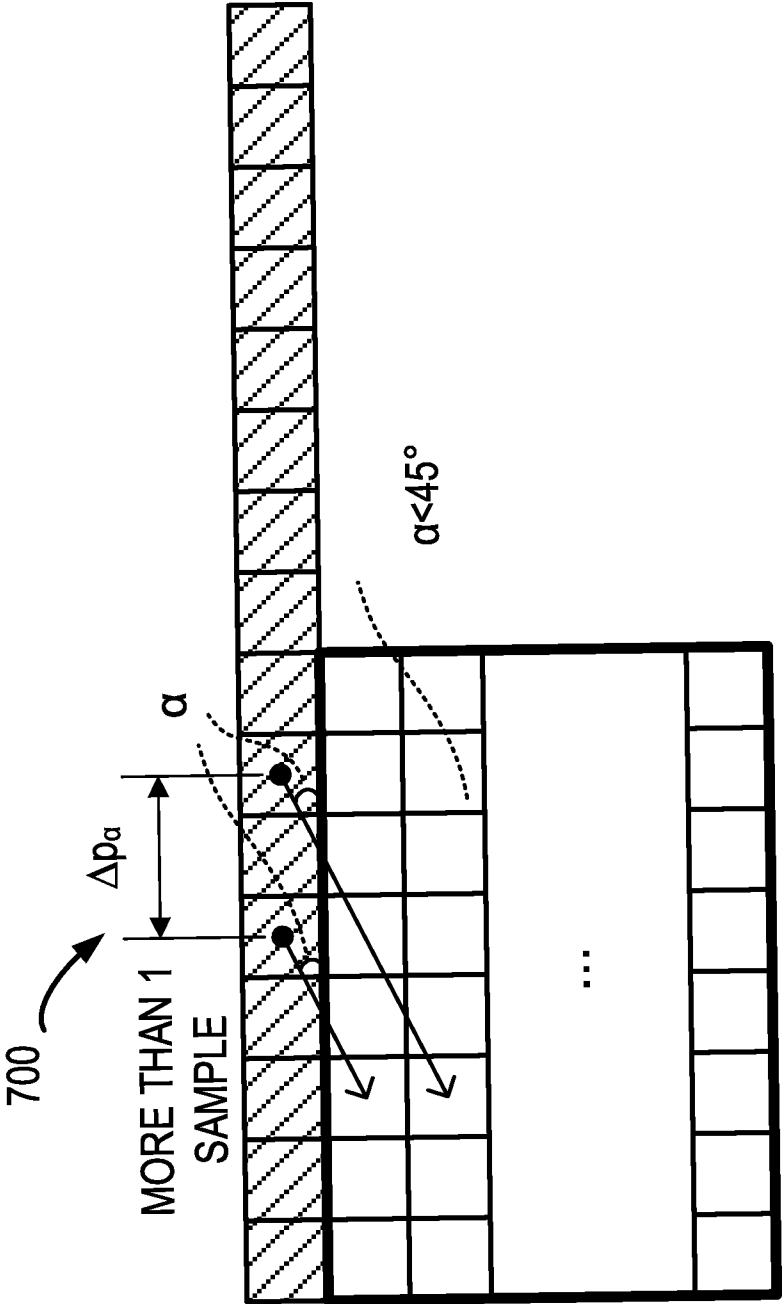


Fig. 7

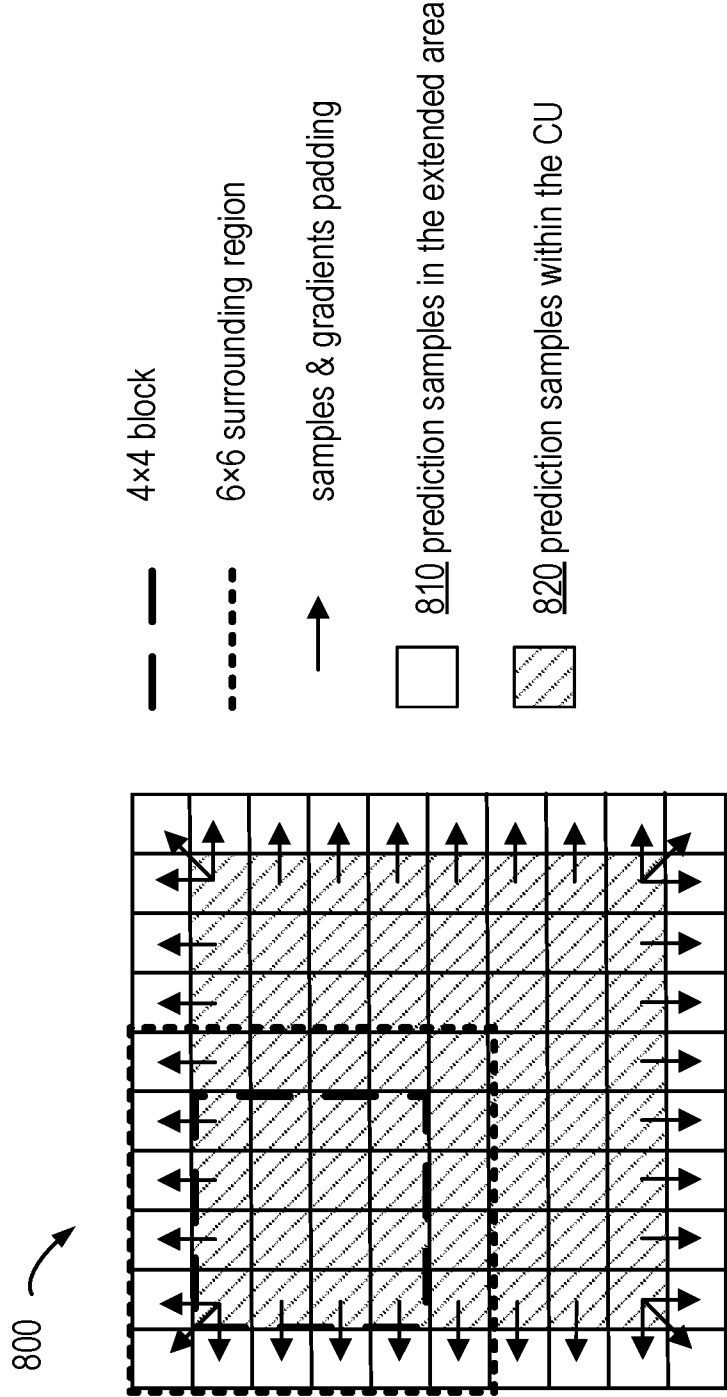


Fig. 8

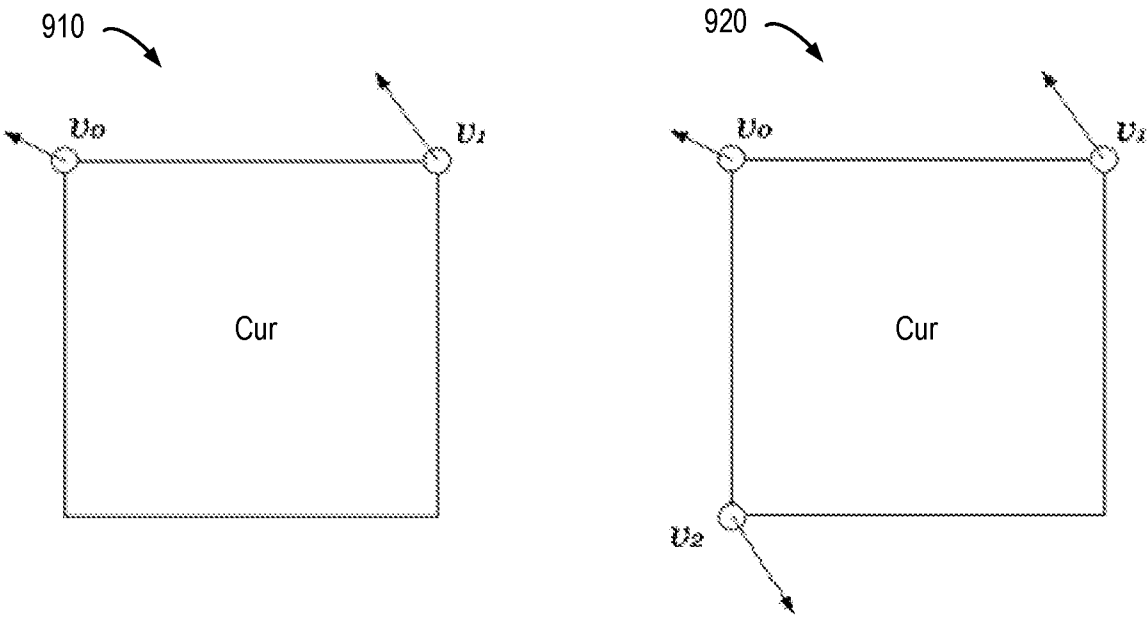


Fig. 9

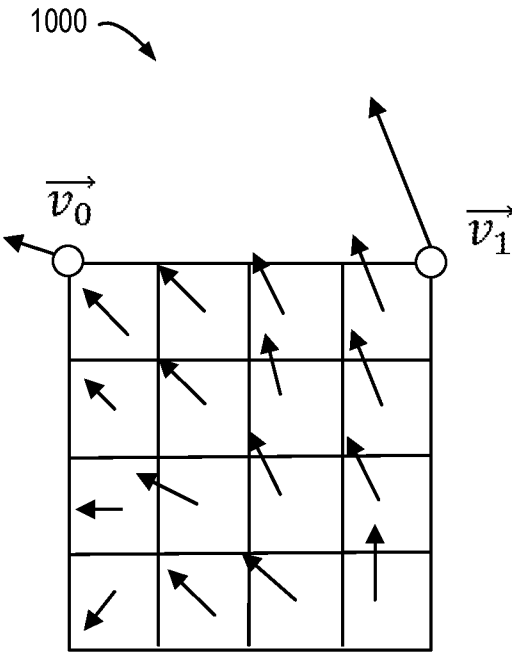


Fig. 10

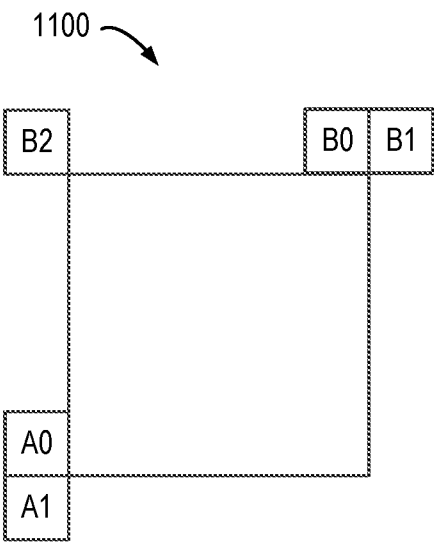


Fig. 11

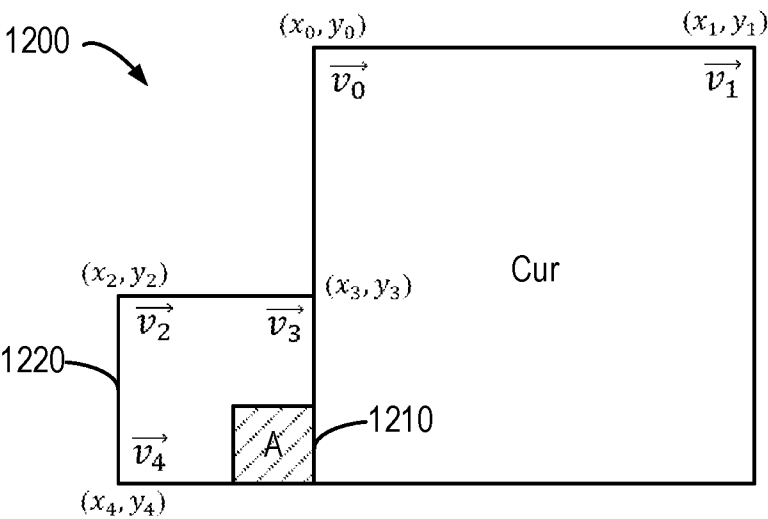


Fig. 12

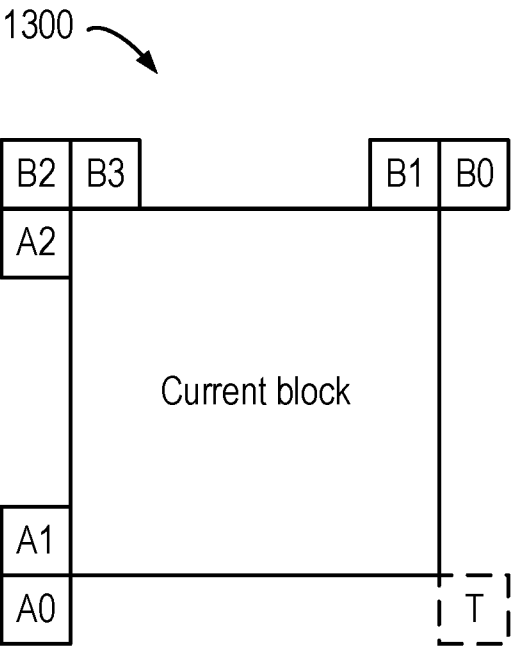


Fig. 13

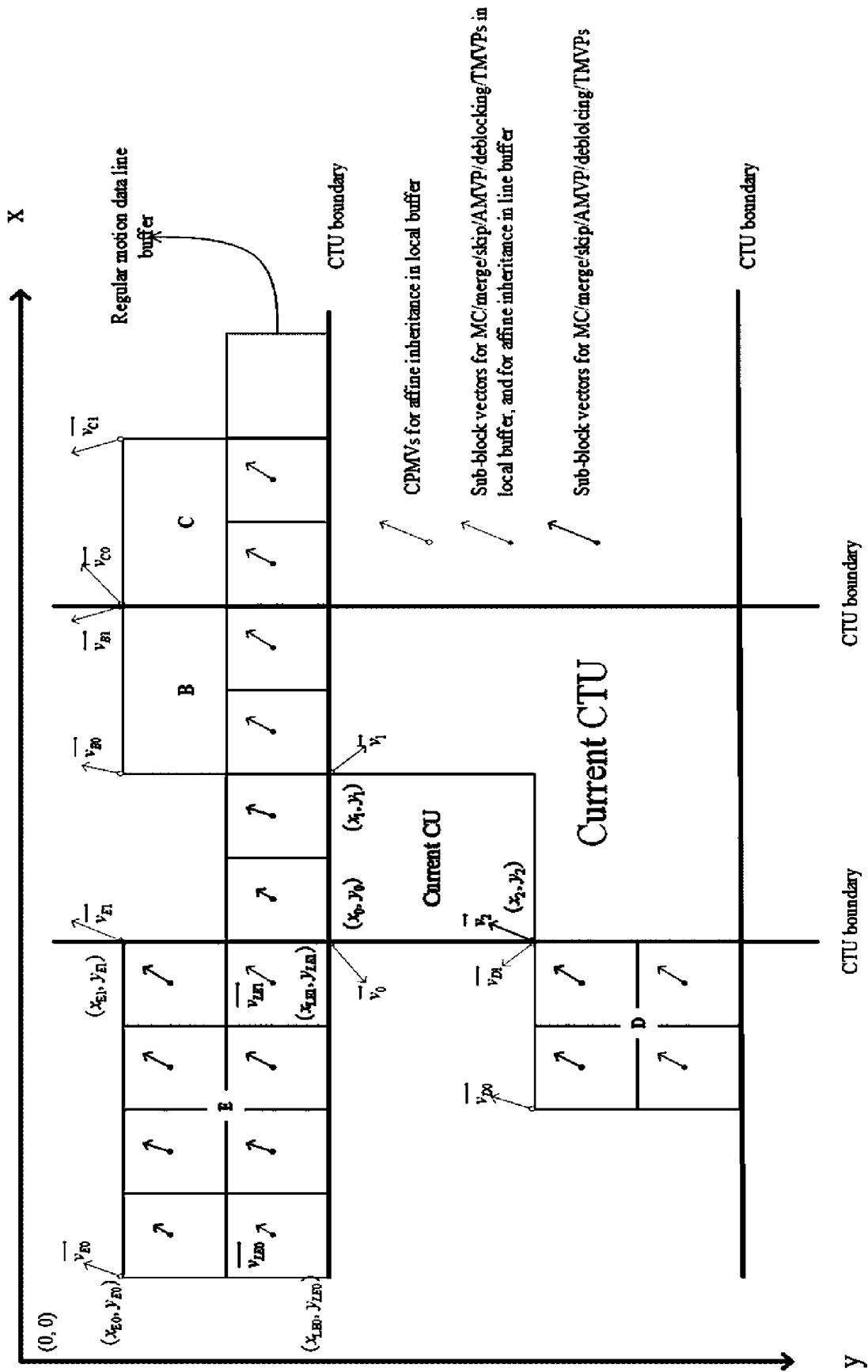


Fig. 14

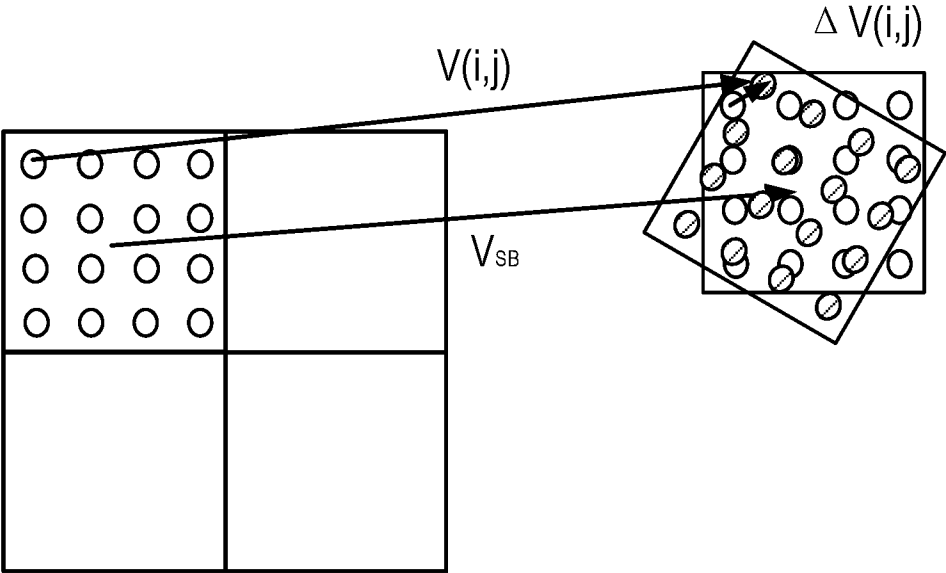


Fig. 15

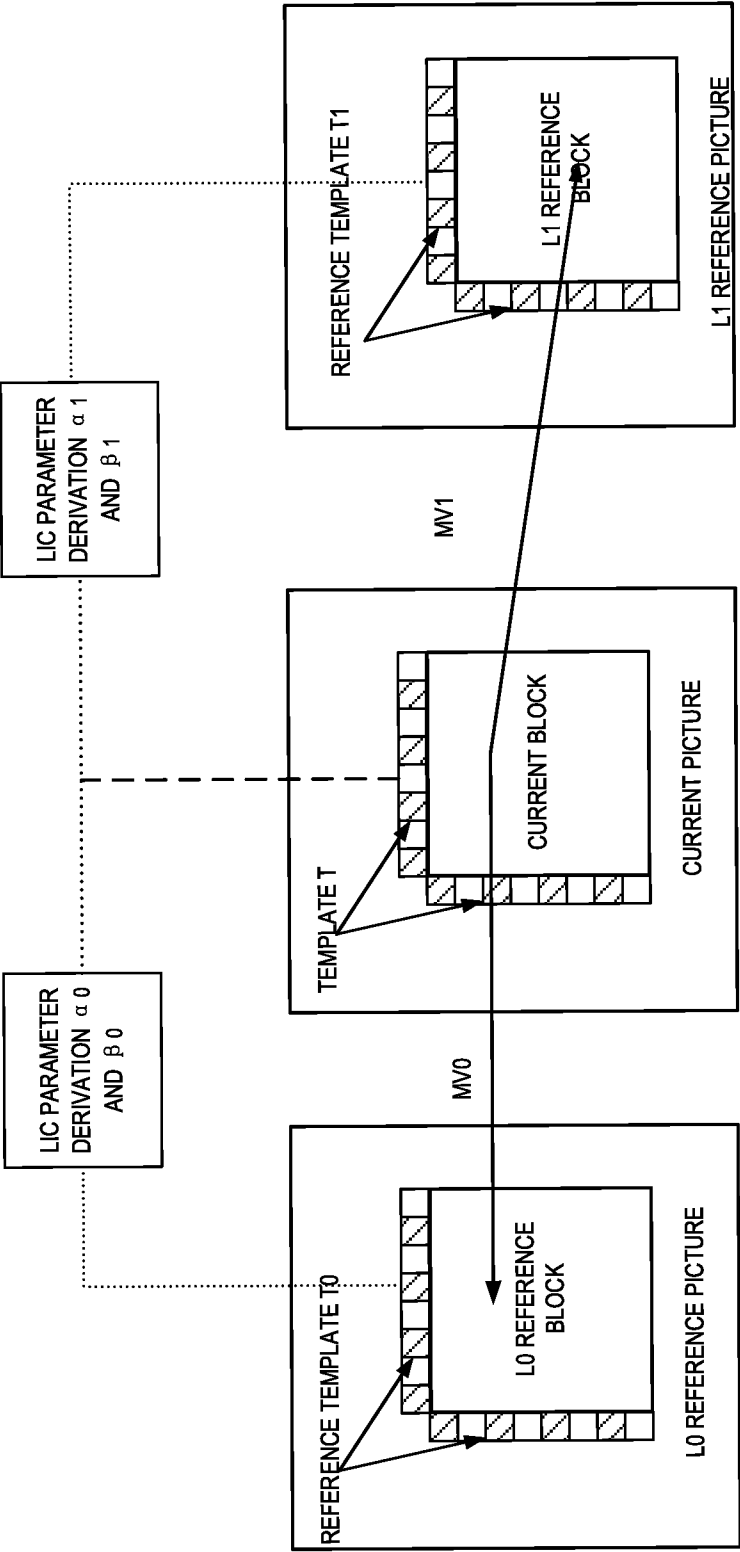


Fig. 16

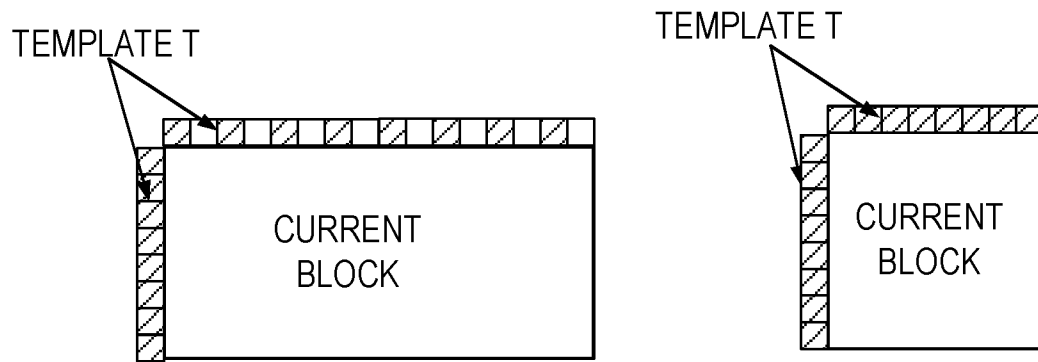


Fig. 17

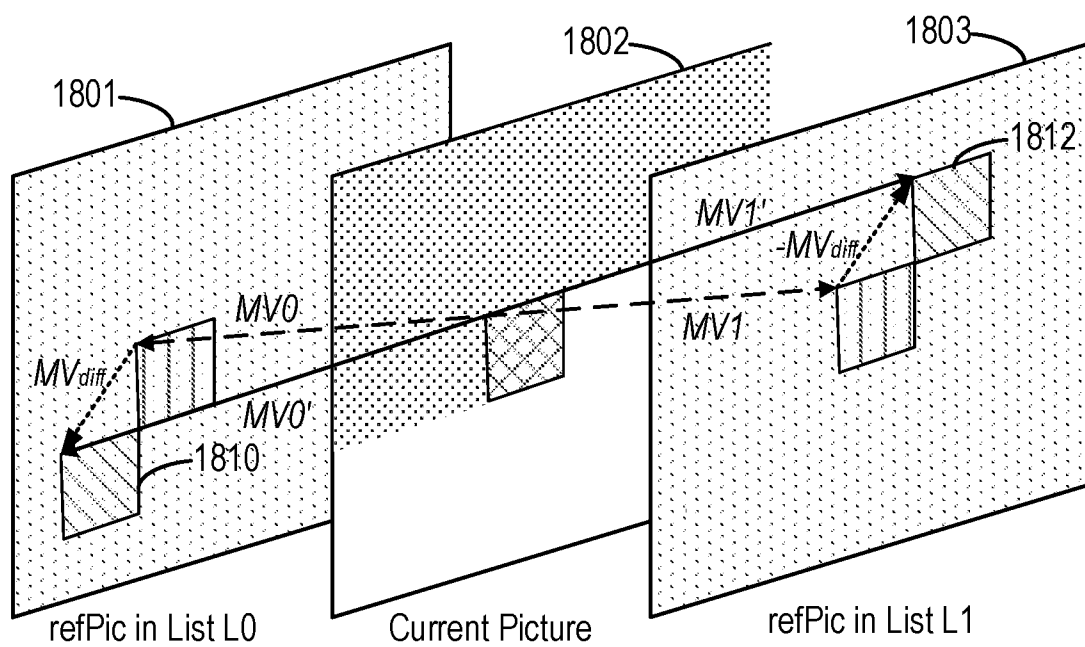


Fig. 18

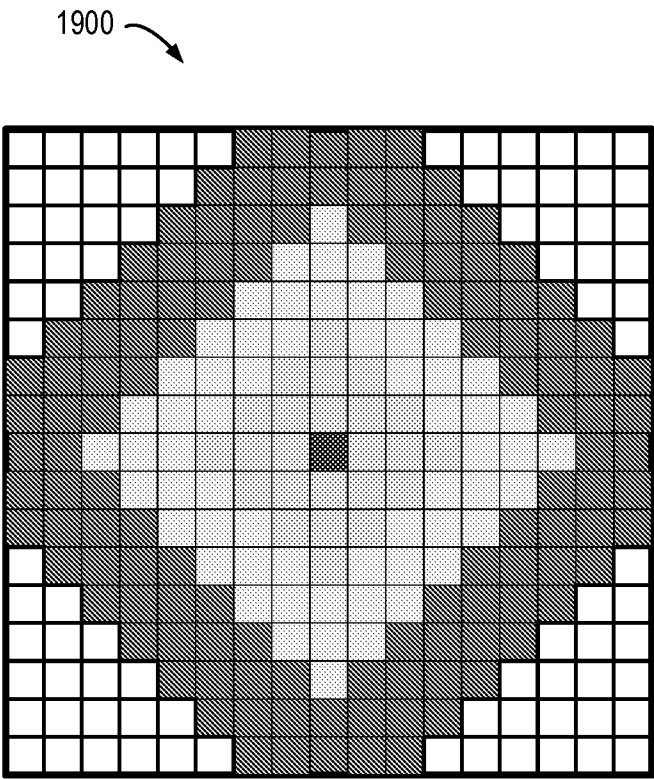


Fig. 19

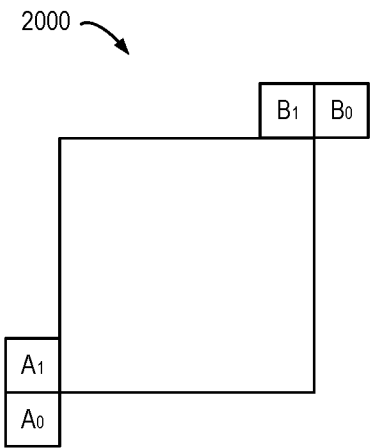


Fig. 20

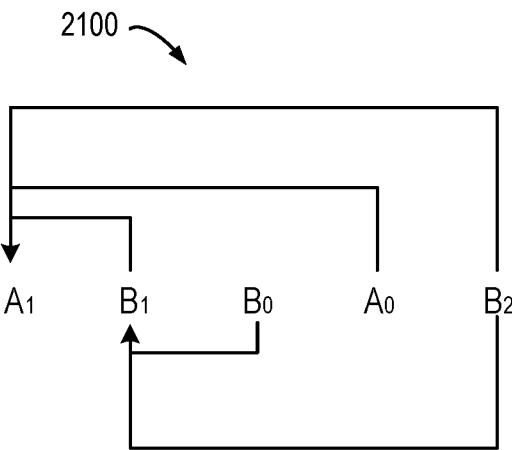


Fig.21

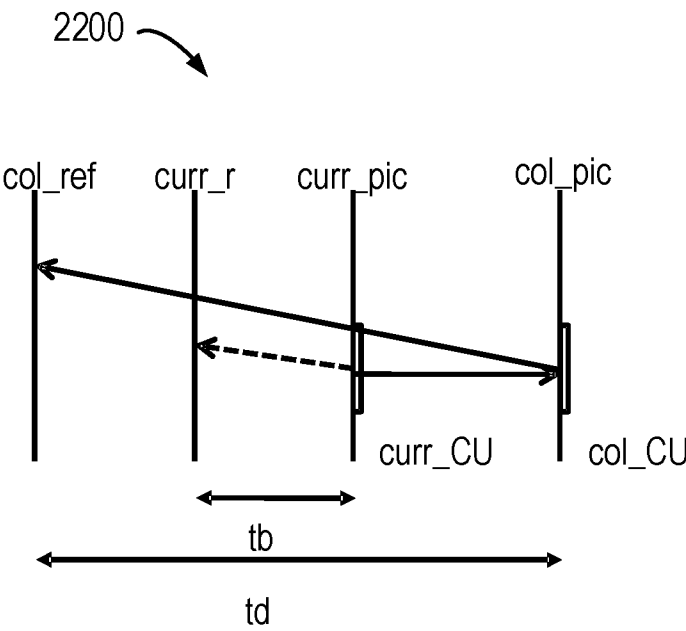


Fig. 22

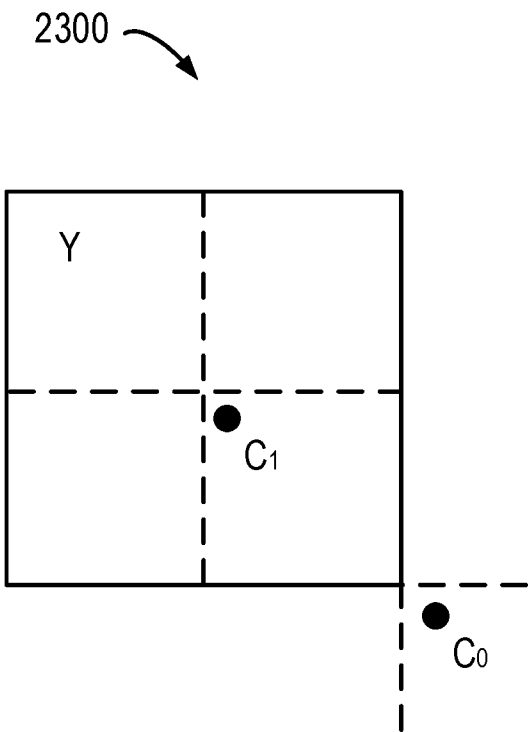


Fig. 23

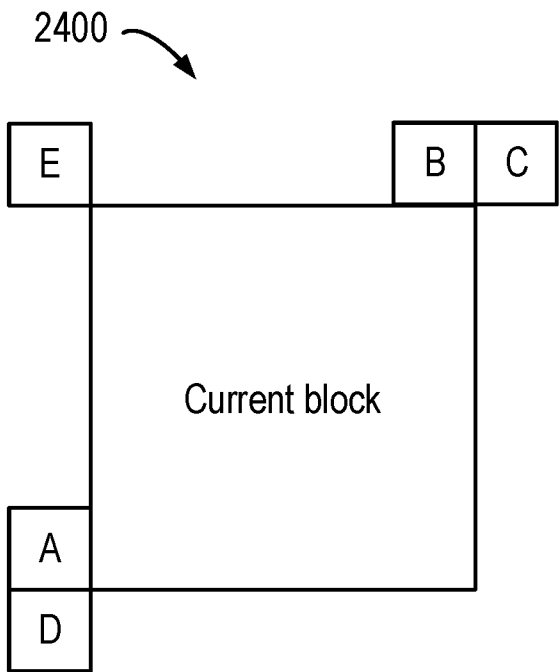


Fig. 24

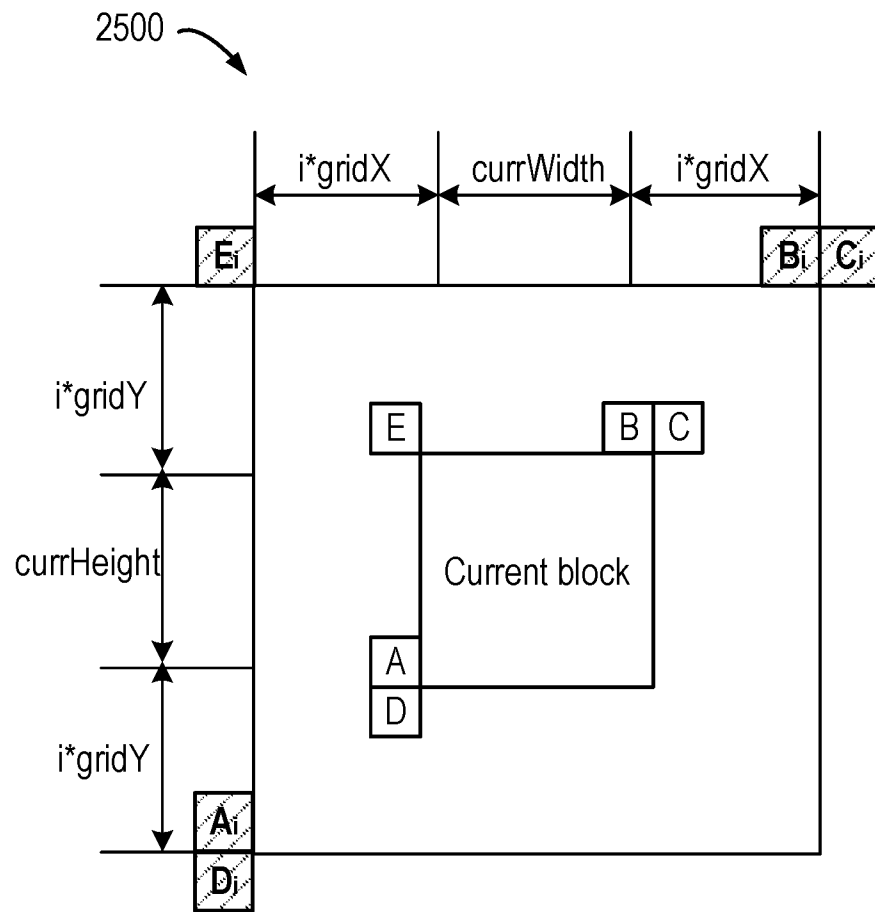


Fig. 25

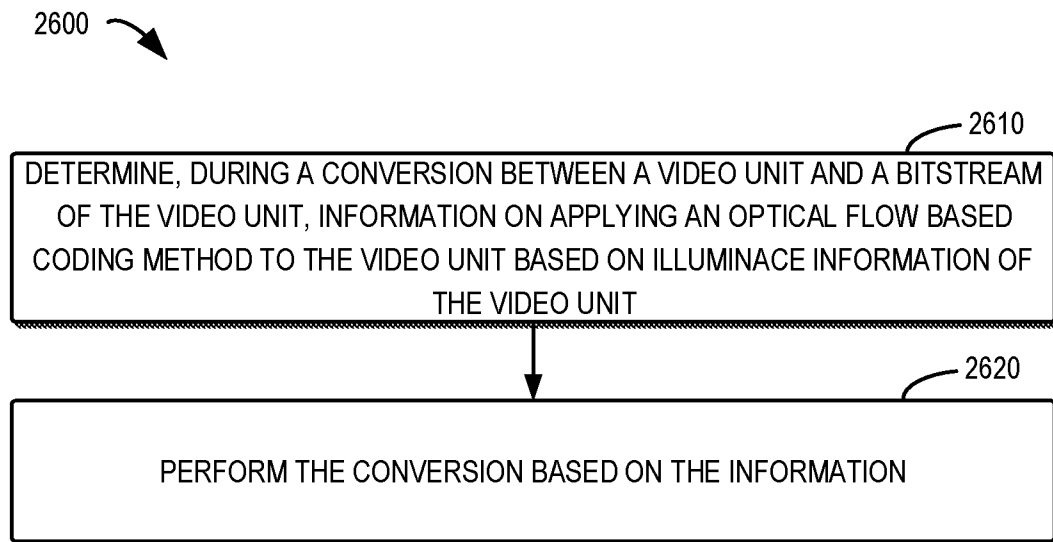
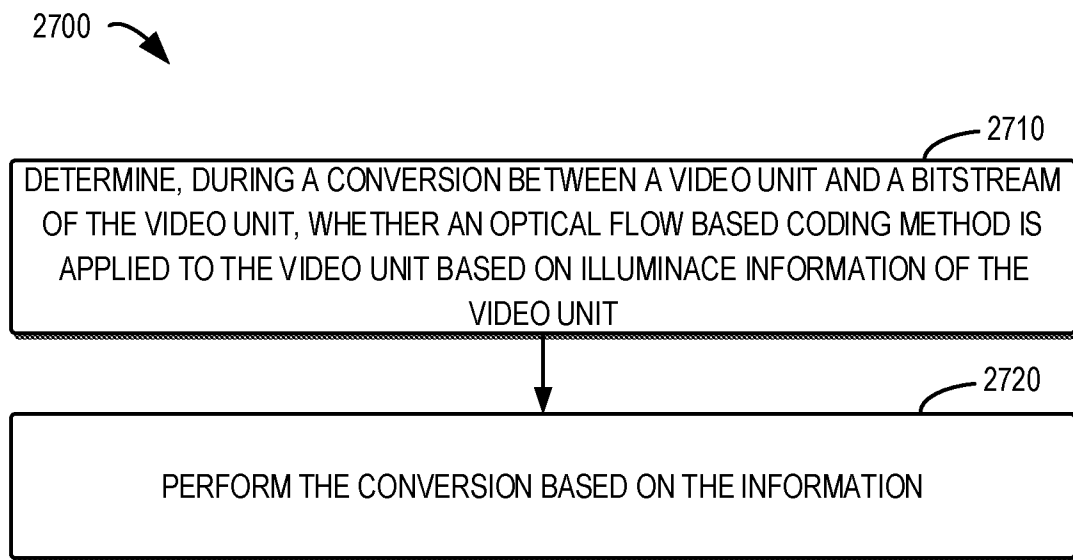
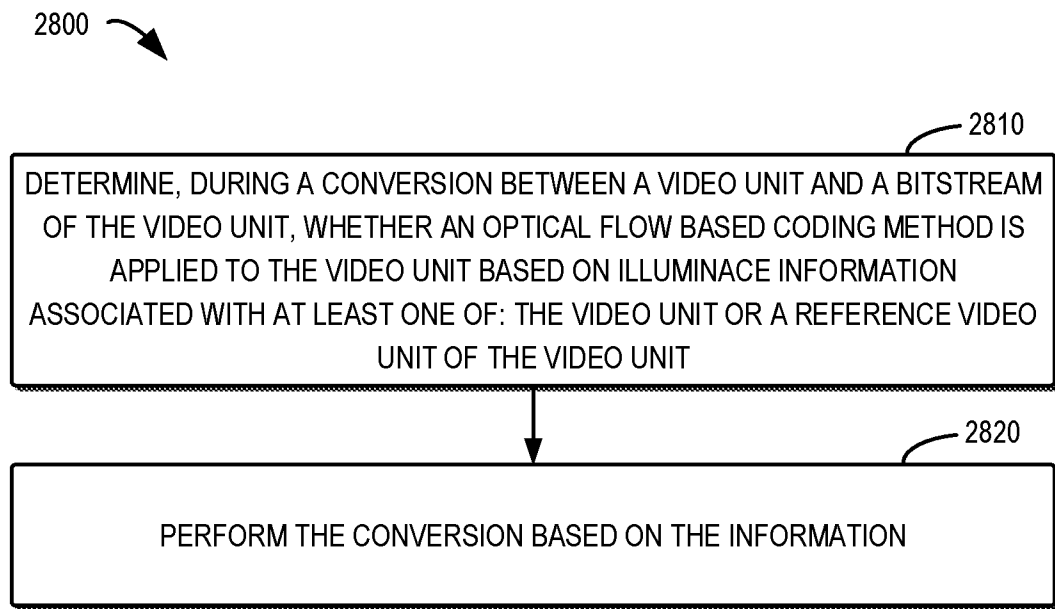
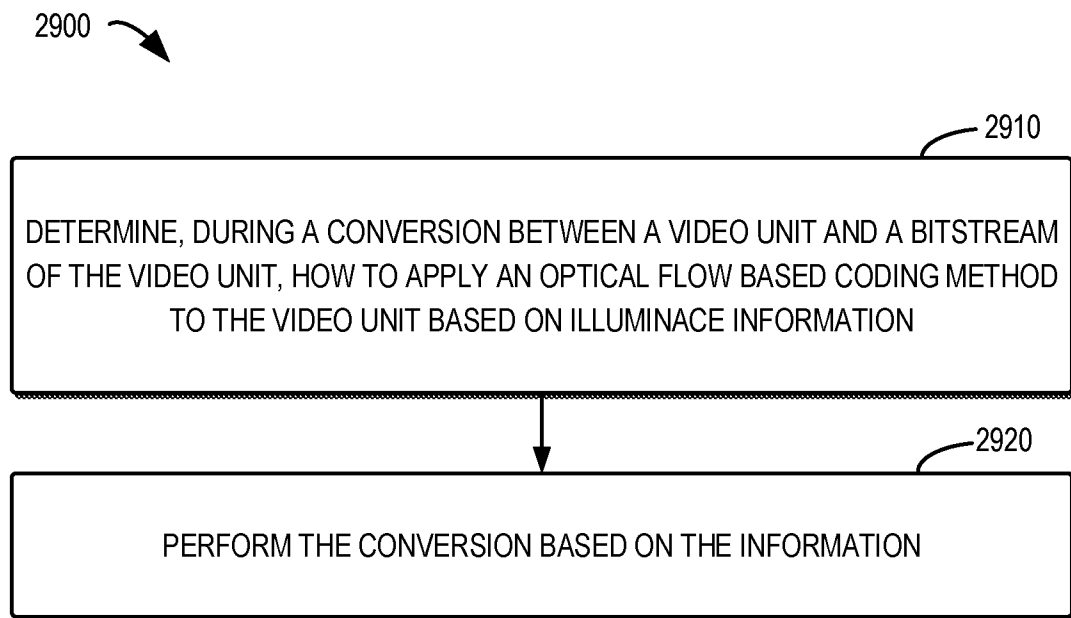
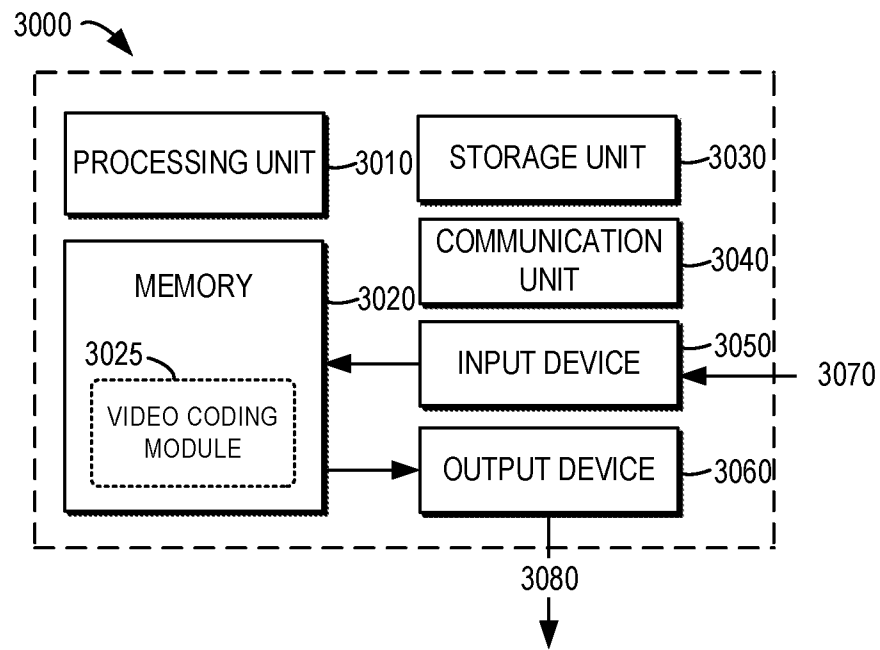


Fig. 26

**Fig. 27**

**Fig. 28**

**Fig. 29**

**Fig. 30**

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2022/097557

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/51(2014.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

JVET,CNKI,CNPAT,WPI,EPODOC: change,difference,illumina nce,luma,chroma,value,flag,SAD, LIC,optical flow,BDOF, PROF,BIO,threshold,disabl+,enabl+,apply+,terminat+,gradient

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	XIU, Xiaoyu et al. "Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 15th Meeting: Gothenburg, SE, 3-12 July 2019" <i>CE4-related: On exclusively applying LIC and BDOF</i> , 12 July 2019 (2019-07-12), Sections 1, 2	1-9, 12-17, 20-22, 25-50
Y	XIU, Xiaoyu et al. "Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 15th Meeting: Gothenburg, SE, 3-12 July 2019" <i>CE4-related: On exclusively applying LIC and BDOF</i> , 12 July 2019 (2019-07-12), Sections 1, 2	10, 11
Y	LIU, Hongbin et al. "Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 15th Meeting: Gothenburg, SE, 3-12 July 2019" <i>Non-CE9: Unified gradient calculations in BDOF</i> , 12 July 2019 (2019-07-12), Section 1	10, 11
A	WO 2017133661 A1 (MEDIATEK INC.) 10 August 2017 (2017-08-10) The whole document	1-50



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered

to be of particular relevance

"E" earlier application or patent but published on or after the international

filing date

"L" document which may throw doubts on priority claim(s) or which is

cited to establish the publication date of another citation or other

special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other

means

"P" document published prior to the international filing date but later than

the priority date claimed

"T" later document published after the international filing date or priority

date and not in conflict with the application but cited to understand the

principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be

considered novel or cannot be considered to involve an inventive step

when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be

considered to involve an inventive step when the document is

combined with one or more other such documents, such combination

being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

28 July 2022

Date of mailing of the international search report

31 August 2022

Name and mailing address of the ISA/CN

National Intellectual Property Administration, PRC
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing
100088, China

Facsimile No. (86-10)62019451

Authorized officer

ZHANG,Luwen

Telephone No. 010-53961724

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2022/097557

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	GAO, Han. "Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/ SC 29/WG 11 16th Meeting: Geneva, CH, 1–11 October 2019" <i>Crosscheck of JVET-P0654: Non-CE4: Unified BDOF and DMVR early termination threshold</i> , 11 October 2019 (2019-10-11), The whole document	1-50
A	CHEN, Ching-Yeh et al. "Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/ IEC JTC 1/SC 29/WG 11 14th Meeting: Geneva, CH, 19–27 March 2019" <i>CE9-related: Simplification of cascading DMVR and BDOF processes</i> , 27 March 2019 (2019-03-27), The whole document	1-50

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2022/097557

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
WO	2017133661	A1	10 August 2017	CN	108781294	A	09 November 2018
				US	2019045215	A1	07 February 2019
				US	2021368204	A1	25 November 2021
				CN	113612993	A	05 November 2021
				EP	3398331	A1	07 November 2018
				IN	201827033012	A	31 July 2020
<hr/>							