(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
02 January 2025 (02.01.2025)

WIPO | PCT

(10) International Publication Number
WO 2025/002276 A1

(51) International Patent Classification:
H04N 19/52 (2014.01)

(21) International Application Number:
PCT/CN2024/102064

(22) International Filing Date:
27 June 2024 (27.06.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
PCT/CN2023/103830
29 June 2023 (29.06.2023) CN
PCT/CN2023/124102
11 October 2023 (11.10.2023) CN
PCT/CN2023/138564
13 December 2023 (13.12.2023) CN

(71) Applicants: **DOUYIN VISION 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, Kai**; 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). **LIU, Hongbin**; 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).

(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, CV, 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, MG, MK, MN, MU, 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,

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

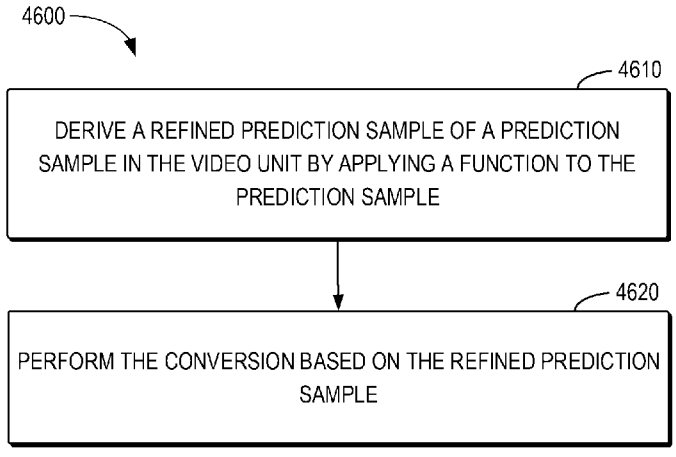

Fig. 46

(57) Abstract: Embodiments of the present disclosure provide a solution for video processing. A method for video processing is proposed. The method comprises: deriving, for a conversion between a video unit of a video and a bitstream of the video, a refined prediction sample of a prediction sample in the video unit by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and performing the conversion based on the refined prediction sample.

WO 2025/002276 A1

[Continued on next page]

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, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, 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, ME, 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).

Published:

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

METHOD, APPARATUS, AND MEDIUM FOR VIDEO PROCESSING

FIELDS

[0001] Embodiments of the present disclosure relates generally to video processing techniques, and more particularly, to local illumination compensation with slope adjustment.

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 video coding techniques is generally expected to be further improved.

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: deriving, for a conversion between a video unit of a video and a bitstream of the video, a refined prediction sample of a prediction sample in the video unit by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and performing the conversion based on the refined prediction sample. In this way, coding performance can be improved by adjusting the parameters.

[0005] In a second aspect, an apparatus for video processing is proposed. The apparatus comprises a processor and a non-transitory memory with instructions thereon. 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, another 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 an apparatus for video processing. The method comprises: deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and generating the bitstream of the video unit based on the refined prediction sample.

[0008] In a fifth aspect, a method for storing a bitstream of a video is proposed. The method comprises: deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; generating the bitstream of the video unit based on the refined prediction sample; 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 illustrates an example of encoder block diagram;

[0015] Fig. 5 illustrates 67 intra prediction modes;

[0016] Fig. 6A and Fig. 6B illustrate reference samples for wide-angular intra prediction, respectively;

[0017] Fig. 7 illustrates problem of discontinuity in case of directions beyond 45°;

[0018] Fig. 8A and Fig. 8B illustrate MMVD Search Point;

[0019] Fig. 9 illustrates an illustration for symmetrical MVD mode;

[0020] Fig. 10 illustrates extended CU region used in BDOF;

[0021] Fig. 11 illustrates top and left neighboring blocks used in CIIP weight derivation;

[0022] Fig. 12A and Fig. 12B illustrate control point based affine motion model;

[0023] Fig. 13 illustrates affine MVF per subblock;

[0024] Fig. 14 illustrates locations of inherited affine motion predictors;

[0025] Fig. 15 illustrates control point motion vector inheritance;

[0026] Fig. 16 illustrates locations of candidates position for constructed affine merge mode;

[0027] Fig. 17 illustrates an illustration of motion vector usage for proposed combined method;

[0028] Fig. 18 illustrates subblock MV VSB and pixel $\Delta v(i,j)$ (red arrow);

[0029] Figs. 19A and 19B illustrate the SbTMVP process in VVC, where Fig. 19A illustrates spatial neighboring blocks used by ATVMP, and Fig. 19B illustrates deriving sub-CU motion field by applying a motion shift from spatial neighbor and scaling the motion information from the corresponding collocated sub-CUs;

[0030] Fig. 20 illustrates local illumination compensation;

[0031] Fig. 21 illustrates no subsampling for the short side;

[0032] Fig. 22 illustrates decoding side motion vector refinement;

[0033] Fig. 23 illustrates diamond regions in the search area;

[0034] Fig. 24 illustrates positions of spatial merge candidate;

[0035] Fig. 25 illustrates candidate pairs considered for redundancy check of spatial merge candidates;

[0036] Fig. 26 illustrates an illustration of motion vector scaling for temporal merge candidate;

[0037] Fig. 27 illustrates candidate positions for temporal merge candidate, C0 and C1;

[0038] Fig. 28 illustrates VVC spatial neighboring blocks of the current block;

[0039] Fig. 29 illustrates an illustration of virtual block in the i-th search round;

[0040] Fig. 30 illustrates examples of the GPM splits grouped by identical angles;

[0041] Fig. 31 illustrates uni-prediction MV selection for geometric partitioning mode;

[0042] Fig. 32 illustrates exemplified generation of a bending weight w0 using geometric partitioning mode;

[0043] Fig. 33 illustrates spatial neighboring blocks used to derive the spatial merge candidates;

[0044] Fig. 34 illustrates template matching performs on a search area around initial MV;

[0045] Fig. 35 illustrates an illustration of sub-blocks where OBMC applies;

[0046] Fig. 36 illustrates SBT position, type and transform type;

[0047] Fig. 37 illustrates neighbouring samples used for calculating SAD;

[0048] Fig. 38 illustrates neighbouring samples used for calculating SAD for sub-CU level motion information;

[0049] Fig. 39 illustrates the sorting process;

[0050] Fig. 40 illustrates reorder process in encoder;

[0051] Fig. 41 illustrates reorder process in decoder;

[0052] Fig. 42 illustrates IBC reference region depending on current CU position;

[0053] Fig. 43 illustrates examples of symmetry in screen content pictures;

[0054] Fig. 44A illustrates an illustration of BV adjustment for horizontal flip;

[0055] Fig. 44B illustrates an illustration of BV adjustment for vertical flip;

[0056] Fig. 45 illustrates intra template matching search area used;

[0057] Fig. 46 illustrates a flowchart of a method for video processing in accordance with embodiments of the present disclosure; and

[0058] Fig. 47 illustrates a block diagram of a computing device in which various embodiments of the present disclosure can be implemented.

[0059] Throughout the drawings, the same or similar reference numerals usually refer to the same or similar elements.

DETAILED DESCRIPTION

[0060] 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.

[0061] 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.

[0062] 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.

[0063] 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.

[0064] 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

[0065] 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.

[0066] 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.

[0067] 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.

[0068] 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.

[0069] 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.

[0070] 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.

[0071] 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.

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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.

[0076] 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.

[0077] 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.

[0078] 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.

[0079] 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.

[0080] 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.

[0081] 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.

[0082] 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.

[0083] 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.

[0084] 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.

[0085] 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.

[0086] 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.

[0087] 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.

[0088] 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.

[0089] 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.

[0090] 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.

[0091] After the reconstruction unit 212 reconstructs the video block, loop filtering operation may be performed to reduce video blocking artifacts in the video block.

[0092] 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.

[0093] 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.

[0094] 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.

[0095] 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.

[0096] 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.

[0097] 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.

[0098] 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.

[0099] 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.

[0100] 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.

[0101] 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 predication and also produces decoded video for presentation on a display device.

[0102] 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. Brief Summary

[0103] The present disclosure is related to video coding technologies. Specifically, it is related to local illumination compensation (LIC), how to improve LIC with slope adjustment, 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. Introduction

[0104] 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.

2.1. Coding flow of a typical video codec

[0105] 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

[0106] 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.

[0107] 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

[0108] 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.

[0109] 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. 6A and Fig. 6B.

[0110] 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

[0111] Fig. 7 shows problem of discontinuity in case of directions beyond 45°. As shown in 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_α. 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.

[0112] 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

[0113] 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)

[0114] 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.

[0115] 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.

[0116] 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.

[0117] In block matching search, the search range is set to cover both the previous and current CTUs.

[0118] 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. Merge mode with MVD (MMVD)

[0119] In addition to merge mode, where the implicitly derived motion information is directly used for prediction samples generation of the current CU, the merge mode with motion vector differences (MMVD) is introduced in VVC. A MMVD flag is signalled right after sending a regular merge flag to specify whether MMVD mode is used for a CU.

[0120] In MMVD, after a merge candidate is selected, it is further refined by the signalled MVDs information. The further information includes a merge candidate flag, an index to specify motion magnitude, and an index for indication of motion direction. In MMVD mode, one for the first two candidates in the merge list is selected to be used as MV basis. The MMVD candidate flag is signalled to specify which one is used between the first and second merge candidates.

[0121] Distance index specifies motion magnitude information and indicate the pre-defined offset from the starting point. As shown in Fig. 8A and Fig. 8B, an offset is added to either horizontal component or vertical component of starting MV. The relation of distance index and pre-defined offset is specified in Table 2.

Table 2 – The relation of distance index and pre-defined offset

Distance IDX	0	1	2	3	4	5	6	7
Offset (in unit of luma sample)	1/4	1/2	1	2	4	8	16	32

[0122] Direction index represents the direction of the MVD relative to the starting point. The direction index can represent of the four directions as shown in Table 3. It's noted that the meaning of MVD sign could be variant according to the information of starting MVs. When the starting MVs is an un-prediction MV or bi-prediction MVs with both lists point to the same side of the current picture (i.e. POCs of two references are both larger than the POC of the current picture, or are both smaller than the POC of the current picture), the sign in Table 3 specifies the sign of MV offset added to the starting MV. When the starting MVs is bi-prediction MVs with the two MVs point to the different sides of the current picture (i.e. the POC of one reference is larger than the POC of the current picture, and the POC of the other reference is smaller than the POC of the current picture), and the difference of POC in list 0 is greater than the one in list 1, the sign in Table 3

specifies the sign of MV offset added to the list0 MV component of starting MV and the sign for the list1 MV has opposite value. Otherwise, if the difference of POC in list 1 is greater than list 0, the sign in Table 3 specifies the sign of MV offset added to the list1 MV component of starting MV and the sign for the list0 MV has opposite value.

[0123] The MVD is scaled according to the difference of POCs in each direction. If the differences of POCs in both lists are the same, no scaling is needed. Otherwise, if the difference of POC in list 0 is larger than the one of list 1, the MVD for list 1 is scaled, by defining the POC difference of L0 as td and POC difference of L1 as tb, described in Fig. 26. If the POC difference of L1 is greater than L0, the MVD for list 0 is scaled in the same way. If the starting MV is uni-predicted, the MVD is added to the available MV.

Table 3 – Sign of MV offset specified by direction index

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

2.6. Symmetric MVD coding

[0124] In VVC, besides the normal unidirectional prediction and bi-directional prediction mode MVD signalling, symmetric MVD mode for bi-predictional MVD signalling is applied. In the symmetric MVD mode, motion information including reference picture indices of both list-0 and list-1 and MVD of list-1 are not signaled but derived.

[0125] The decoding process of the symmetric MVD mode is as follows:

1. At slice level, variables BiDirPredFlag, RefIdxSymL0 and RefIdxSymL1 are derived as follows:
   - If mvd_l1_zero_flag is 1, BiDirPredFlag is set equal to 0.
   - Otherwise, if the nearest reference picture in list-0 and the nearest reference picture in list-1 form a forward and backward pair of reference pictures or a backward and forward pair of reference pictures, BiDirPredFlag is set to 1, and both list-0 and list-1 reference pictures are short-term reference pictures. Otherwise BiDirPredFlag is set to 0.
2. At CU level, a symmetrical mode flag indicating whether symmetrical mode is used or not is explicitly signaled if the CU is bi-prediction coded and BiDirPredFlag is equal to 1.

[0126] When the symmetrical mode flag is true, only mvp_l0_flag, mvp_l1_flag and MVD0 are explicitly signaled. The reference indices for list-0 and list-1 are set equal to the pair of reference pictures, respectively. MVD1 is set equal to (− MVD0). The final

motion vectors are shown in below formula.

$$\begin{cases}(mvx_0, mvy_0) = (mvpx_0 + mvdx_0, mvpy_0 + mvdy_0) \\ (mvx_1, mvy_1) = (mvpx_1 - mvdx_0, mvpy_1 - mvdy_0)\end{cases} \quad (2\text{-}1)$$

[0127] Fig. 9 is an illustration for symmetrical MVD mode. In the encoder, symmetric MVD motion estimation starts with initial MV evaluation. A set of initial MV candidates comprising of the MV obtained from uni-prediction search, the MV obtained from bi-prediction search and the MVs from the AMVP list. The one with the lowest rate-distortion cost is chosen to be the initial MV for the symmetric MVD motion search.

2.7. Bi-directional optical flow (BDOF)

[0128] 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.

[0129] 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.

[0130] 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 4x4 subblock. The following steps are applied in the BDOF process.

[0131] 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.,

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

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 shift1 = max(6, bitDepth-6).

[0132] Then, the auto- and cross-correlation of the gradients, S_1, S_2, S_3, S_5 and S_6, are calculated as

$$\begin{aligned}S_1 = \textstyle\sum_{(i,j)\in\Omega} Abs(\psi_x(i,j)),\ S_3 = \textstyle\sum_{(i,j)\in\Omega} \theta(i,j)\cdot Sign(\psi_x(i,j))\\ S_2 = \sum_{(i,j)\in\Omega} \psi_x(i,j)\cdot Sign(\psi_y(i,j))\\ S_5 = \textstyle\sum_{(i,j)\in\Omega} Abs(\psi_y(i,j)),\ S_6 = \textstyle\sum_{(i,j)\in\Omega} \theta(i,j)\cdot Sign(\psi_y(i,j))\end{aligned} \tag{2-3}$$

where

$$\begin{aligned}\psi_x(i,j) &= \left(\frac{\partial I^{(1)}}{\partial x}(i,j) + \frac{\partial I^{(0)}}{\partial x}(i,j)\right) \gg n_a\\ \psi_y(i,j) &= \left(\frac{\partial I^{(1)}}{\partial y}(i,j) + \frac{\partial I^{(0)}}{\partial y}(i,j)\right) \gg n_a\\ \theta(i,j) &= \left(I^{(1)}(i,j) \gg n_b\right) - \left(I^{(0)}(i,j) \gg n_b\right)\end{aligned} \tag{2-4}$$

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, bitDepth − 11) and min(4, bitDepth − 8), respectively.

[0133] The motion refinement (v_x, v_y) is then derived using the cross- and auto-correlation terms using the following:

$$\begin{aligned}v_x &= S_1 > 0?\ clip3\left(-th'_{BIO}, th'_{BIO}, -\left((S_3 \cdot 2^{n_b-n_a}) \gg \lfloor \log_2 S_1 \rfloor\right)\right) : 0\\ v_y &= S_5 > 0?\ clip3\left(-th'_{BIO}, th'_{BIO}, -\left(\left(S_6 \cdot 2^{n_b-n_a}\right.\right.\right.\\ &\qquad \left.\left.\left. - \left((v_x S_{2,m}) \ll n_{S_2} + v_x S_{2,s}\right)/2\right) \gg \lfloor \log_2 S_5 \rfloor\right)\right) : 0\end{aligned} \tag{2-5}$$

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 $n_{S_2} = 12$.

[0134] Based on the motion refinement and the gradients, the following adjustment is calculated for each sample in the 4×4 subblock:

$$b(x,y) = rnd\left(\left(v_x\left(\frac{\partial I^{(1)}(x,y)}{\partial x} - \frac{\partial I^{(0)}(x,y)}{\partial x}\right) + v_y\left(\frac{\partial I^{(1)}(x,y)}{\partial y} - \frac{\partial I^{(0)}(x,y)}{\partial y}\right) + 1\right)/2\right). \quad (2\text{-}6)$$

[0135] Finally, the BDOF samples of the CU are calculated by adjusting the bi-prediction samples as follows:

$$pred_{BDOF}(x,y) = \left(I^{(0)}(x,y) + I^{(1)}(x,y) + b(x,y) + o_{offset}\right) \gg shift. \quad (2\text{-}7)$$

[0136] 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.

[0137] 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. 10 shows extended CU region used in BDOF. As depicted in Fig. 10, 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 (white positions) 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 (gray positions). 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.

[0138] 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 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.

[0139] 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.

2.8. Combined inter and intra prediction (CIIP)

[0140] In VVC, when a CU is coded in merge mode, if the CU contains at least 64 luma samples (that is, CU width times CU height is equal to or larger than 64), and if both CU width and CU height are less than 128 luma samples, an additional flag is signalled to indicate if the combined inter/intra prediction (CIIP) mode is applied to the current CU. As its name indicates, the CIIP prediction combines an inter prediction signal with an intra prediction signal. The inter prediction signal in the CIIP mode P_{inter} is derived using the same inter prediction process applied to regular merge mode; and the intra prediction signal P_{intra} is derived following the regular intra prediction process with the planar mode. Then, the intra and inter prediction signals are combined using weighted averaging, where the weight value is calculated depending on the coding modes of the top and left neighbouring blocks (depicted in Fig. 11) as follows:

- If the top neighbor is available and intra coded, then set isIntraTop to 1, otherwise set isIntraTop to 0;
- If the left neighbor is available and intra coded, then set isIntraLeft to 1, otherwise set isIntraLeft to 0;
- If (isIntraLeft + isIntraTop) is equal to 2, then wt is set to 3;
- Otherwise, if (isIntraLeft + isIntraTop) is equal to 1, then wt is set to 2;
- Otherwise, set wt to 1.

[0141] The CIIP prediction is formed as follows:

$$P_{\text{CIIP}} = \left((4 - wt) * P_{inter} + wt * P_{intra} + 2\right) \gg 2. \tag{2-8}$$

2.9. Affine motion compensated prediction

[0142] 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. Fig. 12A and Fig. 12B show control point based affine motion model, where Fig. 12A shows 4-parameter affine model and Fig. 12B shows 6-parameter affine model. As shown Fig. 12A and Fig. 12B , 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).

[0143] For 4-parameter affine motion model, motion vector at sample location (x, y) in a block is derived as:

$$\begin{cases} mv_x = \frac{mv_{1x}-mv_{0x}}{W}x + \frac{mv_{1y}-mv_{0y}}{W}y + mv_{0x} \\ mv_y = \frac{mv_{1y}-mv_{0y}}{W}x + \frac{mv_{1y}-mv_{0x}}{W}y + mv_{0y} \end{cases} \quad (2\text{-}9).$$

[0144] For 6-parameter affine motion model, motion vector at sample location (x, y) in a block is derived as:

$$\begin{cases} mv_x = \frac{mv_{1x}-mv_{0x}}{W}x + \frac{mv_{2x}-mv_{0x}}{H}y + mv_{0x} \\ mv_y = \frac{mv_{1y}-mv_{0y}}{W}x + \frac{mv_{2y}-mv_{0y}}{H}y + mv_{0y} \end{cases} \quad (2\text{-}10).$$

[0145] 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.

[0146] In order to simplify the motion compensation prediction, block based affine transform prediction is applied. To derive motion vector of each 4×4 luma subblock, the motion vector of the center sample of each subblock, as shown in Fig. 13, 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.

[0147] As done for translational motion inter prediction, there are also two affine motion inter prediction modes: affine merge mode and affine AMVP mode.

2.9.1. Affine merge prediction

[0148] 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.

[0149] 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. 14 shows locations of inherited affine motion

predictors. The candidate blocks are shown in Fig. 14. For the left predictor, the scan order is A0->A1, and for the above predictor, the scan order is B0->B1->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. As shown in , if the neighbour left bottom block A 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 which contains the block A are attained. When block A 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. Fig. 15 shows control point motion vector inheritance.

[0150] Constructed affine candidate means the candidate is constructed by combining the neighbour translational motion information of each control point. Fig. 16 shows locations of candidates position for constructed affine merge mode. The motion information for the control points is derived from the specified spatial neighbours and temporal neighbour shown in Fig. 16. $CPMV_k$ (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.

[0151] 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:

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

[0152] 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.

[0153] 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.

2.9.2. Affine AMVP prediction

[0154] 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 AVMP candidate list size is 2 and it is generated by using the following four types of CPVM 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.

[0155] 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 AVMP 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.

[0156] Constructed AMVP candidate is derived from the specified spatial neighbours shown in Fig. 16. 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.

[0157] 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.

2.9.3. Affine motion information storage

[0158] 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.

[0159] 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. Fig. 17 is an illustration of motion vector usage for proposed combined method. As shown in Fig. 17, 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.

2.9.4. Prediction refinement with optical flow for affine mode

[0160] 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:

[0161] Step 1) The subblock-based affine motion compensation is performed to generate subblock prediction $I(i,j)$.

[0162] 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.

$$g_x(i,j) = (I(i+1,j) \gg shift1) - (I(i-1,j) \gg shift1) \quad (2\text{-}11)$$

$$g_y(i,j) = (I(i,j+1) \gg shift1) - (I(i,j-1) \gg shift1) \quad (2\text{-}12)$$

$shift1$ 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.

[0163] Step 3) The luma prediction refinement is calculated by the following optical flow equation.

$$\Delta I(i,j) = g_x(i,j) * \Delta v_x(i,j) + g_y(i,j) * \Delta v_y(i,j) \tag{2-13}$$

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. 18. The $\Delta v(i,j)$ is quantized in the unit of 1/32 luam sample precision.

[0164] 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,

$$\begin{cases} \mathrm{dx}(i,j) = i - x_{SB} \\ \mathrm{dy}(i,j) = j - y_{SB} \end{cases} \tag{2-14}$$

$$\begin{cases} \Delta v_x(i,j) = C * dx(i,j) + D * dy(i,j) \\ \Delta v_y(i,j) = E * dx(i,j) + F * dy(i,j) \end{cases} \tag{2-15}$$

[0165] In order to keep accuracy, the enter 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.

[0166] For 4-parameter affine model,

$$\begin{cases} C = F = \frac{v_{1x} - v_{0x}}{w} \\ E = -D = \frac{v_{1y} - v_{0y}}{w} \end{cases} \tag{2-16}$$

[0167] For 6-parameter affine model,

$$\begin{cases} C = \frac{v_{1x} - v_{0x}}{w} \\ D = \frac{v_{2x} - v_{0x}}{h} \\ E = \frac{v_{1y} - v_{0y}}{w} \\ F = \frac{v_{2y} - v_{0y}}{h} \end{cases} \tag{2-17}$$

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.

[0168] 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.

$$I'(i,j) = I(i,j) + \Delta I(i,j) \quad (2\text{-}18)$$

[0169] 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.

[0170] 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.

2.10. Subblock-based temporal motion vector prediction (SbTMVP)

[0171] VVC supports the subblock-based temporal motion vector prediction (SbTMVP) method. Similar to the temporal motion vector prediction (TMVP) in HEVC, SbTMVP uses the motion field in the collocated picture to improve motion vector prediction and merge mode for CUs in the current picture. The same collocated picture used by TMVP is used for SbTVMP. SbTMVP differs from TMVP in the following two main aspects:

- TMVP predicts motion at CU level, but SbTMVP predicts motion at sub-CU level;
- Whereas TMVP fetches the temporal motion vectors from the collocated block in the collocated picture (the collocated block is the bottom-right or center block relative to the current CU), SbTMVP applies a motion shift before fetching the temporal motion information from the collocated picture, where the motion shift is obtained from the motion vector from one of the spatial neighboring blocks of the current CU.

[0172] The SbTVMP process is illustrated in Fig. 19A and Fig. 19B. SbTMVP predicts the motion vectors of the sub-CUs within the current CU in two steps. In the first step, the spatial neighbor A1 in Fig. 19A is examined. If A1 has a motion vector that uses the

collocated picture as its reference picture, this motion vector is selected to be the motion shift to be applied. If no such motion is identified, then the motion shift is set to (0, 0).

[0173] In the second step, the motion shift identified in Step 1 is applied (i.e., added to the current block's coordinates) to obtain sub-CU-level motion information (motion vectors and reference indices) from the collocated picture as shown in Fig. 19B. The example in Fig. 19B assumes the motion shift is set to block A1's motion. Then, for each sub-CU, the motion information of its corresponding block (the smallest motion grid that covers the center sample) in the collocated picture is used to derive the motion information for the sub-CU. After the motion information of the collocated sub-CU is identified, it is converted to the motion vectors and reference indices of the current sub-CU in a similar way as the TMVP process of HEVC, where temporal motion scaling is applied to align the reference pictures of the temporal motion vectors to those of the current CU.

[0174] In VVC, a combined subblock based merge list which contains both SbTVMP candidate and affine merge candidates is used for the signalling of subblock based merge mode. The SbTVMP mode is enabled/disabled by a sequence parameter set (SPS) flag. If the SbTMVP mode is enabled, the SbTMVP predictor is added as the first entry of the list of subblock based merge candidates, and followed by the affine merge candidates. The size of subblock based merge list is signalled in SPS and the maximum allowed size of the subblock based merge list is 5 in VVC.

[0175] The sub-CU size used in SbTMVP is fixed to be 8x8, and as done for affine merge mode, SbTMVP mode is only applicable to the CU with both width and height are larger than or equal to 8.

[0176] The encoding logic of the additional SbTMVP merge candidate is the same as for the other merge candidates, that is, for each CU in P or B slice, an additional RD check is performed to decide whether to use the SbTMVP candidate.

2.11. Adaptive motion vector resolution (AMVR)

[0177] In HEVC, motion vector differences (MVDs) (between the motion vector and predicted motion vector of a CU) are signalled in units of quarter-luma-sample when use_integer_mv_flag is equal to 0 in the slice header. In VVC, a CU-level adaptive motion vector resolution (AMVR) scheme is introduced. AMVR allows MVD of the CU to be coded in different precision. Dependent on the mode (normal AMVP mode or affine AVMP mode) for the current CU, the MVDs of the current CU can be adaptively selected

as follows:

- Normal AMVP mode: quarter-luma-sample, half-luma-sample, integer-luma-sample or four-luma-sample.
- Affine AMVP mode: quarter-luma-sample, integer-luma-sample or 1/16 luma-sample.

[0178] The CU-level MVD resolution indication is conditionally signalled if the current CU has at least one non-zero MVD component. If all MVD components (that is, both horizontal and vertical MVDs for reference list L0 and reference list L1) are zero, quarter-luma-sample MVD resolution is inferred.

[0179] For a CU that has at least one non-zero MVD component, a first flag is signalled to indicate whether quarter-luma-sample MVD precision is used for the CU. If the first flag is 0, no further signaling is needed and quarter-luma-sample MVD precision is used for the current CU. Otherwise, a second flag is signalled to indicate half-luma-sample or other MVD precisions (integer or four-luma sample) is used for normal AMVP CU. In the case of half-luma-sample, a 6-tap interpolation filter instead of the default 8-tap interpolation filter is used for the half-luma sample position. Otherwise, a third flag is signalled to indicate whether integer-luma-sample or four-luma-sample MVD precision is used for normal AMVP CU. In the case of affine AMVP CU, the second flag is used to indicate whether integer-luma-sample or 1/16 luma-sample MVD precision is used. In order to ensure the reconstructed MV has the intended precision (quarter-luma-sample, half-luma-sample, integer-luma-sample or four-luma-sample), the motion vector predictors for the CU will be rounded to the same precision as that of the MVD before being added together with the MVD. The motion vector predictors are rounded toward zero (that is, a negative motion vector predictor is rounded toward positive infinity and a positive motion vector predictor is rounded toward negative infinity).

[0180] The encoder determines the motion vector resolution for the current CU using RD check. To avoid always performing CU-level RD check four times for each MVD resolution, in VTM11, the RD check of MVD precisions other than quarter-luma-sample is only invoked conditionally. For normal AVMP mode, the RD cost of quarter-luma-sample MVD precision and integer-luma sample MV precision is computed first. Then, the RD cost of integer-luma-sample MVD precision is compared to that of quarter-luma-sample MVD precision to decide whether it is necessary to further check the RD cost of four-luma-sample MVD precision. When the RD cost for quarter-luma-sample MVD precision is much smaller than that of the integer-luma-sample MVD precision, the RD

check of four-luma-sample MVD precision is skipped. Then, the check of half-luma-sample MVD precision is skipped if the RD cost of integer-luma-sample MVD precision is significantly larger than the best RD cost of previously tested MVD precisions. For affine AMVP mode, if affine inter mode is not selected after checking rate-distortion costs of affine merge/skip mode, merge/skip mode, quarter-luma-sample MVD precision normal AMVP mode and quarter-luma-sample MVD precision affine AMVP mode, then 1/16 luma-sample MV precision and 1-pel MV precision affine inter modes are not checked. Furthermore, affine parameters obtained in quarter-luma-sample MV precision affine inter mode is used as starting search point in 1/16 luma-sample and quarter-luma-sample MV precision affine inter modes.

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

[0181] 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.

$$P_{\text{bi-pred}} = \left((8 - w) * P_0 + w * P_1 + 4\right) \gg 3 \tag{2-19}$$

[0182] 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 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∈{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. 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.

[0183] 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.

[0184] 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.

[0185] 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.

2.13. Local illumination compensation (LIC)

[0186] 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:

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

[0187] 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. 20 illustrates the LIC process. In Fig. 20, 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. 20) and their corresponding reference samples in the temporal reference pictures (i.e., either *T0* or *T1* in Fig. 20). 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. 20 are used to derive α and β.

[0188] To improve the coding performance, no subsampling for the short side is performed as shown in Fig. 21.

2.14. Decoder side motion vector refinement (DMVR)

[0189] 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. As illustrated in Fig. 22, the SAD between the two blocks based on each MV candidate (e.g., MV0' and MV1') around the initial MV is calculated. The MV candidate with the lowest SAD becomes the refined MV and used to generate the bi-predicted signal.

[0190] 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.

[0191] 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.

[0192] The additional features of DMVR are mentioned in the following sub-clauses.

2.14.1. Searching scheme

[0193] 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:

$$MV0' = MV0 + MV\_offset \quad (2\text{-}20)$$

$$MV1' = MV1 - MV\_offset \quad (2\text{-}21)$$

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.

[0194] 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.

[0195] 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.

[0196] 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

$$E(x, y) = A(x - x_{min})^2 + B(y - y_{min})^2 + C \quad (2\text{-}22)$$

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:

$$x_{min} = (E(-1,0) - E(1,0))/(2(E(-1,0) + E(1,0) - 2E(0,0))) \quad (2\text{-}23)$$

$$y_{min} = (E(0,-1) - E(0,1))/(2((E(0,-1) + E(0,1) - 2E(0,0)))) \quad (2\text{-}24)$$

[0197] 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 peal 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.

2.14.2. Bilinear-interpolation and sample padding

[0198] 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 DVMR 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.

2.14.3. Maximum DMVR processing unit

[0199] 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.

2.15. Multi-pass decoder-side motion vector refinement

[0200] 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.

2.15.1. First pass – Block based bilateral matching MV refinement

[0201] 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.

[0202] BM performs local search to derive integer sample precision intDeltaMV and half-pel sample precision halfDeltaMv. The local search applies a 3×3 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.

[0203] The bilateral matching cost is calculated as: bilCost = mvDistanceCost + sadCost. When the block size cbW * 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.

[0204] The existing fractional sample refinement is further applied to derive the final deltaMV. The refined MVs after the first pass are then derived as:

- MV0_pass1 = MV0 + deltaMV.
- MV1_pass1 = MV1 – deltaMV.

2.15.2. Second pass – Subblock based bilateral matching MV refinement

[0205] 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 (MV0_pass2(sbIdx2) and MV1_pass2(sbIdx2)) are derived based on the minimum bilateral matching cost between the two reference subblocks in L0

and L1.

[0206] For each subblock, BM performs full search to derive integer sample precision intDeltaMV. The full search has a 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.

[0207] The bilateral matching cost is calculated by applying a cost factor to the SATD cost between the two reference subblocks, as: bilCost = satdCost * costFactor. The search area (2*sHor + 1) * (2*sVer + 1) is divided up to 5 diamond shape search regions shown on Fig. 23. 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.

[0208] 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.

[0209] The existing VVC DMVR fractional sample refinement is further applied to derive the final deltaMV(sbIdx2). The refined MVs at second pass is then derived as:

- MV0_pass2(sbIdx2) = MV0_pass1 + deltaMV(sbIdx2).
- MV1_pass2(sbIdx2) = MV1_pass1 – deltaMV(sbIdx2).

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

[0210] 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 Vx and Vy without clipping starting from the refined MV of the parent subblock of the second pass. The derived bioMv(Vx, Vy) is rounded to 1/16 sample precision and clipped between -32 and 32.

[0211] The refined MVs (MV0_pass3(sbIdx3) and MV1_pass3(sbIdx3)) at third pass are derived as:

- MV0_pass3(sbIdx3) = MV0_pass2(sbIdx2) + bioMv.
- MV1_pass3(sbIdx3) = MV0_pass2(sbIdx2) – bioMv.

2.16. Sample-based BDOF

[0212] In the sample-based BDOF, instead of deriving motion refinement (Vx, Vy) on a block basis, it is performed per sample.

[0213] 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 Vx and Vy. The derived motion refinement (Vx, Vy) is applied to adjust the bi-predicted sample value for the center sample of the window.

2.17. Extended merge prediction

[0214] 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.

[0215] 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.

[0216] 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.

2.17.1. Spatial candidates derivation

[0217] Fig. 24 shows positions of spatial merge candidate. The derivation of spatial merge candidates in VVC is same to that in HEVC except the positions of first two merge candidates are swapped. A maximum of four merge candidates are selected among candidates located in the positions depicted in . The order of derivation is B0, A0, B1, A1 and B2. Position B2 is considered only when one or more than one CUs of position B0, A0, B1, A1 are not available (e.g. because it belongs to another slice or tile) or is intra

coded. After candidate at position A1 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. Instead only the pairs linked with an arrow in Fig. 25 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.

2.17.2. Temporal candidates derivation

[0218] 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 Fig. 26, 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.

[0219] The position for the temporal candidate is selected between candidates C0 and C1, as depicted in Fig. 27. If CU at position C0 is not available, is intra coded, or is outside of the current row of CTUs, position C1 is used. Otherwise, position C0 is used in the derivation of the temporal merge candidate.

2.17.3. History-based merge candidates derivation

[0220] 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.

[0221] 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.

[0222] To reduce the number of redundancy check operations, the following simplifications are introduced:

[0223] Number of HMPV candidates is used for merge list generation is set as (N <= 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.

[0224] 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.

2.17.4. Pair-wise average merge candidates derivation

[0225] 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.

[0226] 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.

2.17.5. Merge estimation region

[0227] 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 (xCb + cbWidth) >> Log2ParMrgLevel is greater than xCb >> Log2ParMrgLevel and (yCb + cbHeight) >> Log2ParMrgLevel is great than (yCb >> Log2ParMrgLevel) and where (xCb, yCb) 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.

2.18. New merge candidates

2.18.1. Non-adjacent merge candidates derivation

[0228] In VVC, five spatially neighboring blocks shown in Fig. 28 as well as one temporal neighbor are used to derive merge candidates.

[0229] 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:

[0230] First, the relative position of the virtual block to the current block is calculated by:

Offsetx =-i×gridX, Offsety = -i×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.

[0231] Second, the width and height of the virtual block are calculated by:

newWidth = i×2×gridX+ currWidth newHeight = i×2×gridY + 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.

[0232] gridX and gridY are currently set to currWidth and currHeight, respectively.

[0233] Fig. 29 illustrates the relationship between the virtual block and the current block.

[0234] 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.

[0235] 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.

[0236] Non-adjacent spatial merge candidates are inserted into the merge list after the temporal merge candidate in the order of B_1->A_1->C_1->D_1->E_1.

2.18.2. STMVP

[0237] It is proposed to derive an averaging candidate as STMVP candidate using three spatial merge candidates and one temporal merge candidate.

[0238] STMVP is inserted before the above-left spatial merge candidate.

[0239] The STMVP candidate is pruned with all the previous merge candidates in the merge list.

[0240] For the spatial candidates, the first three candidates in the current merge candidate list are used.

[0241] For the temporal candidate, the same position as VTM / HEVC collocated position is used.

[0242] 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.

[0243] The temporal candidate with the same position as VTM / HEVC collocated position used in TMVP is denoted as Col.

[0244] 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 × 3 + mvLX_S × 3 + mvLX_Col × 2) >>3 or

mvLX = (mvLX_F × 3 + mvLX_T × 3 + mvLX_Col × 2) >>3 or

mvLX = (mvLX_S × 3 + mvLX_T × 3 + mvLX_Col × 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 or

mvLX = (mvLX_S + mvLX_Col) >>1 or

mvLX = (mvLX_T + mvLX_Col) >>1

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

2.18.3. Merge list size

[0245] 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.

2.19. Geometric partitioning mode (GPM)

[0246] In VVC, a geometric partitioning mode is supported for inter prediction. The geometric partitioning mode is signalled using a CU-level flag as one kind of merge mode, with other merge modes including the regular merge mode, the MMVD mode, the CIIP mode and the subblock merge mode. In total 64 partitions are supported by geometric partitioning mode for each possible CU size $w \times h = 2^m \times 2^n$ with $m, n \in \{3 \cdots 6\}$ excluding 8x64 and 64x8.

[0247] When this mode is used, a CU is split into two parts by a geometrically located straight line (Fig. 30). Fig. 30 shows examples of the GPM splits grouped by identical angles. The location of the splitting line is mathematically derived from the angle and offset parameters of a specific partition. Each part of a geometric partition in the CU is inter-predicted using its own motion; only uni-prediction is allowed for each partition, that is, each part has one motion vector and one reference index. The uni-prediction motion constraint is applied to ensure that same as the conventional bi-prediction, only two motion compensated prediction are needed for each CU. The uni-prediction motion for each partition is derived using the process described in 2.19.1.

[0248] If geometric partitioning mode is used for the current CU, then a geometric partition index indicating the partition mode of the geometric partition (angle and offset), and two merge indices (one for each partition) are further signalled. The number of maximum GPM candidate size is signalled explicitly in SPS and specifies syntax binarization for GPM merge indices. After predicting each of part of the geometric

partition, the sample values along the geometric partition edge are adjusted using a blending processing with adaptive weights as in 2.19.2. This is the prediction signal for the whole CU, and transform and quantization process will be applied to the whole CU as in other prediction modes. Finally, the motion field of a CU predicted using the geometric partition modes is stored as in 2.19.3.

2.19.1. Uni-prediction candidate list construction

[0249] The uni-prediction candidate list is derived directly from the merge candidate list constructed according to the extended merge prediction process in 2.17. Denote n as the index of the uni-prediction motion in the geometric uni-prediction candidate list. The LX motion vector of the n-th extended merge candidate, with X equal to the parity of n, is used as the n-th uni-prediction motion vector for geometric partitioning mode. These motion vectors are marked with "x" in Fig. 31. In case a corresponding LX motion vector of the n-the extended merge candidate does not exist, the L(1 − X) motion vector of the same candidate is used instead as the uni-prediction motion vector for geometric partitioning mode.

2.19.2. Blending along the geometric partitioning edge

[0250] After predicting each part of a geometric partition using its own motion, blending is applied to the two prediction signals to derive samples around geometric partition edge. The blending weight for each position of the CU are derived based on the distance between individual position and the partition edge.

[0251] The distance for a position (x, y) to the partition edge are derived as:

$$d(x,y) = (2x + 1 - w)\cos(\varphi_i) + (2y + 1 - h)\sin(\varphi_i) - \rho_j \quad (2\text{-}25)$$

$$\rho_j = \rho_{x,j}\cos(\varphi_i) + \rho_{y,j}\sin(\varphi_i) \quad (2\text{-}26)$$

$$\rho_{x,j} = \begin{cases} 0 & i\,\%\,16 = 8 \text{ or } (i\,\%\,16 \neq 0 \text{ and } h \geq w) \\ \pm(j \times w) \gg 2 & otherwise \end{cases} \quad (2\text{-}27)$$

$$\rho_{y,j} = \begin{cases} \pm(j \times h) \gg 2 & i\,\%\,16 = 8 \text{ or } (i\,\%\,16 \neq 0 \text{ and } h \geq w) \\ 0 & otherwise \end{cases} \quad (2\text{-}28)$$

where i, j are the indices for angle and offset of a geometric partition, which depend on the signaled geometric partition index. The sign of $\rho_{x,j}$ and $\rho_{y,j}$ depend on angle index i.

[0252] The weights for each part of a geometric partition are derived as following:

$$wIdxL(x,y) = partIdx\ ?\ 32 + d(x,y) : 32 - d(x,y) \quad (2\text{-}29)$$

$$w_0(x,y) = \frac{Clip3(0,8,(wIdxL(x,y)+4)\gg 3)}{8} \quad (2\text{-}30)$$

$$w_1(x,y) = 1 - w_0(x,y) \quad (2\text{-}31)$$

[0253] The partIdx depends on the angle index i. One example of weigh w_0 is illustrated in Fig. 32. Fig. 32 shows exemplified generation of a bending weight $\boldsymbol{w_0}$ using geometric partitioning mode.

2.19.3. Motion field storage for geometric partitioning mode

[0254] Mv1 from the first part of the geometric partition, Mv2 from the second part of the geometric partition and a combined Mv of Mv1 and Mv2 are stored in the motion filed of a geometric partitioning mode coded CU.

[0255] The stored motion vector type for each individual position in the motion filed are determined as:

$$sType = abs(motionIdx) < 32\ ?\ 2 : (motionIdx \leq 0\ ?\ (1 - partIdx) : partIdx) \quad (2\text{-}32)$$

[0256] where motionIdx is equal to $d(4x+2, 4y+2)$, which is recalculated from equation (2-18). The partIdx depends on the angle index i.

[0257] If sType is equal to 0 or 1, Mv0 or Mv1 are stored in the corresponding motion field, otherwise if sType is equal to 2, a combined Mv from Mv0 and Mv2 are stored. The combined Mv are generated using the following process:

1) If Mv1 and Mv2 are from different reference picture lists (one from L0 and the other from L1), then Mv1 and Mv2 are simply combined to form the bi-prediction motion vectors.

Otherwise, if Mv1 and Mv2 are from the same list, only uni-prediction motion Mv2 is stored.

2.20. Multi-hypothesis prediction

[0258] In multi-hypothesis prediction (MHP), up to two additional predictors are signalled on top of inter AMVP mode, regular merge mode, affine merge and MMVD mode. The resulting overall prediction signal is accumulated iteratively with each additional prediction signal.

$$p_{n+1} = (1 - \alpha_{n+1})p_n + \alpha_{n+1}h_{n+1}$$

[0259] The weighting factor α is specified according to the following Table 4:

Table 4 – weighting factor for MHP

add_hyp_weight_idx	α
0	1/4

1	-1/8

[0260] For inter AMVP mode, MHP is only applied if non-equal weight in BCW is selected in bi-prediction mode.

[0261] The additional hypothesis can be either merge or AMVP mode. In the case of merge mode, the motion information is indicated by a merge index, and the merge candidate list is the same as in the Geometric Partition Mode. In the case of AMVP mode, the reference index, MVP index, and MVD are signaled.

2.21. Non-adjacent spatial candidate

[0262] The non-adjacent spatial merge candidates are inserted after the TMVP in the regular merge candidate list. The pattern of the spatial merge candidates is shown on Fig. 33. The distances between the non-adjacent spatial candidates and the current coding block are based on the width and height of the current coding block.

2.22. Template matching (TM)

[0263] Template matching (TM) is a decoder-side MV derivation method to refine the motion information of the current CU by finding the closest match between a template (i.e., top and/or left neighbouring blocks of the current CU) in the current picture and a block (i.e., same size to the template) in a reference picture. As illustrated in Fig. 34, a better MV is to be searched around the initial motion of the current CU within a [– 8, +8]-pel search range. The template matching in this contribution has two modifications: search step size is determined based on AMVR mode and TM can be cascaded with bilateral matching process in merge modes.

[0264] In AMVP mode, an MVP candidate is determined based on template matching error to pick up the one which reaches the minimum difference between current block template and reference block template, and then TM performs only for this particular MVP candidate for MV refinement. TM refines this MVP candidate, starting from full-pel MVD precision (or 4-pel for 4-pel AMVR mode) within a [–8, +8]-pel search range by using iterative diamond search. The AMVP candidate may be further refined by using cross search with full-pel MVD precision (or 4-pel for 4-pel AMVR mode), followed sequentially by half-pel and quarter-pel ones depending on AMVR mode as specified in Table 5. This search process ensures that the MVP candidate still keeps the same MV precision as indicated by AMVR mode after TM process.

Table 5 – Search patterns of AMVR and merge mode with AMVR.

Search pattern	AMVR mode				Merge mode	
	4-pel	Full-pel	Half-pel	Quarter-pel	AltIF=0	AltIF=1
4-pel diamond	v					
4-pel cross	v					
Full-pel diamond		v	v	v	v	v
Full-pel cross		v	v	v	v	v
Half-pel cross			v	v	v	v
Quarter-pel cross				v	v	
1/8-pel cross					v	

[0265] In merge mode, similar search method is applied to the merge candidate indicated by the merge index. As Table 5 shows, TM may perform all the way down to 1/8-pel MVD precision or skipping those beyond half-pel MVD precision, depending on whether the alternative interpolation filter (that is used when AMVR is of half-pel mode) is used according to merged motion information. Besides, when TM mode is enabled, template matching may work as an independent process or an extra MV refinement process between block-based and subblock-based bilateral matching (BM) methods, depending on whether BM can be enabled or not according to its enabling condition check.

2.23. Overlapped block motion compensation (OBMC)

[0266] Overlapped Block Motion Compensation (OBMC) has previously been used in H.263. In the JEM, unlike in H.263, OBMC can be switched on and off using syntax at the CU level. When OBMC is used in the JEM, the OBMC is performed for all motion compensation (MC) block boundaries except the right and bottom boundaries of a CU. Moreover, it is applied for both the luma and chroma components. In the JEM, a MC block is corresponding to a coding block. When a CU is coded with sub-CU mode (includes sub-CU merge, affine and FRUC mode), each sub-block of the CU is a MC block. To process CU boundaries in a uniform fashion, OBMC is performed at sub-block level for all MC block boundaries, where sub-block size is set equal to 4×4, as illustrated in Fig. 35.

[0267] When OBMC applies to the current sub-block, besides current motion vectors, motion vectors of four connected neighbouring sub-blocks, if available and are not

identical to the current motion vector, are also used to derive prediction block for the current sub-block. These multiple prediction blocks based on multiple motion vectors are combined to generate the final prediction signal of the current sub-block.

[0268] Prediction block based on motion vectors of a neighbouring sub-block is denoted as P_N, with N indicating an index for the neighbouring *above*, *below*, *left* and *right* sub-blocks and prediction block based on motion vectors of the current sub-block is denoted as P_C. When P_N is based on the motion information of a neighbouring sub-block that contains the same motion information to the current sub-block, the OBMC is not performed from P_N. Otherwise, every sample of P_N is added to the same sample in P_C, i.e., four rows/columns of P_N are added to P_C. The weighting factors {1/4, 1/8, 1/16, 1/32} are used for P_N and the weighting factors {3/4, 7/8, 15/16, 31/32} are used for P_C. The exception are small MC blocks, (i.e., when height or width of the coding block is equal to 4 or a CU is coded with sub-CU mode), for which only two rows/columns of P_N are added to P_C. In this case weighting factors {1/4, 1/8} are used for P_N and weighting factors {3/4, 7/8} are used for P_C. For P_N generated based on motion vectors of vertically (horizontally) neighbouring sub-block, samples in the same row (column) of P_N are added to P_C with a same weighting factor.

[0269] In the JEM, for a CU with size less than or equal to 256 luma samples, a CU level flag is signalled to indicate whether OBMC is applied or not for the current CU. For the CUs with size larger than 256 luma samples or not coded with AMVP mode, OBMC is applied by default. At the encoder, when OBMC is applied for a CU, its impact is taken into account during the motion estimation stage. The prediction signal formed by OBMC using motion information of the top neighbouring block and the left neighbouring block is used to compensate the top and left boundaries of the original signal of the current CU, and then the normal motion estimation process is applied.

2.24. Multiple transform selection (MTS) for core transform

[0270] In addition to DCT-II which has been employed in HEVC, a Multiple Transform Selection (MTS) scheme is used for residual coding both inter and intra coded blocks. It uses multiple selected transforms from the DCT8/DST7. The newly introduced transform matrices are DST-VII and DCT-VIII. Table 6 shows the basis functions of the selected DST/DCT.

Table 6 – Transform basis functions of DCT-II/ VIII and DSTVII for N-point input

Transform Type	Basis function $T_i(j)$, $i, j = 0, 1, \ldots, N-1$
DCT-II	$T_i(j) = \omega_0 \cdot \sqrt{\frac{2}{N}} \cdot \cos\left(\frac{\pi \cdot i \cdot (2j+1)}{2N}\right)$ where, $\omega_0 = \begin{cases} \sqrt{\frac{2}{N}} & i = 0 \\ 1 & i \neq 0 \end{cases}$
DCT-VIII	$T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \cos\left(\frac{\pi \cdot (2i+1) \cdot (2j+1)}{4N+2}\right)$
DST-VII	$T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \sin\left(\frac{\pi \cdot (2i+1) \cdot (j+1)}{2N+1}\right)$

[0271] In order to keep the orthogonality of the transform matrix, the transform matrices are quantized more accurately than the transform matrices in HEVC. To keep the intermediate values of the transformed coefficients within the 16-bit range, after horizontal and after vertical transform, all the coefficients are to have 10-bit.

[0272] In order to control MTS scheme, separate enabling flags are specified at SPS level for intra and inter, respectively. When MTS is enabled at SPS, a CU level flag is signalled to indicate whether MTS is applied or not. Here, MTS is applied only for luma. The MTS signaling is skipped when one of the below conditions is applied.

- The position of the last significant coefficient for the luma TB is less than 1 (i.e., DC only).
- The last significant coefficient of the luma TB is located inside the MTS zero-out region.

[0273] If MTS CU flag is equal to zero, then DCT2 is applied in both directions. However, if MTS CU flag is equal to one, then two other flags are additionally signalled to indicate the transform type for the horizontal and vertical directions, respectively. Transform and signalling mapping table as shown in Table 7. Unified the transform selection for ISP and implicit MTS is used by removing the intra-mode and block-shape dependencies. If current block is ISP mode or if the current block is intra block and both intra and inter explicit MTS is on, then only DST7 is used for both horizontal and vertical transform cores. When it comes to transform matrix precision, 8-bit primary transform cores are used. Therefore, all the transform cores used in HEVC are kept as the same, including 4-point DCT-2 and DST-7, 8-point, 16-point and 32-point DCT-2. Also, other transform cores including 64-point DCT-2, 4-point DCT-8, 8-point, 16-point, 32-point

DST-7 and DCT-8, use 8-bit primary transform cores.

Table 7 – Transform and signalling mapping table

MTS_CU_flag	MTS_Hor_flag	MTS_Ver_flag	Intra/inter	
			Horizontal	Vertical
0			DCT2	
1	0	0	DST7	DST7
	0	1	DCT8	DST7
	1	0	DST7	DCT8
	1	1	DCT8	DCT8

[0274] To reduce the complexity of large size DST-7 and DCT-8, High frequency transform coefficients are zeroed out for the DST-7 and DCT-8 blocks with size (width or height, or both width and height) equal to 32. Only the coefficients within the 16x16 lower-frequency region are retained.

[0275] As in HEVC, the residual of a block can be coded with transform skip mode. To avoid the redundancy of syntax coding, the transform skip flag is not signalled when the CU level MTS_CU_flag is not equal to zero. Note that implicit MTS transform is set to DCT2 when LFNST or MIP is activated for the current CU. Also the implicit MTS can be still enabled when MTS is enabled for inter coded blocks.

2.25. Subblock transform (SBT)

[0276] In VTM, subblock transform is introduced for an inter-predicted CU. In this transform mode, only a sub-part of the residual block is coded for the CU. When inter-predicted CU with cu_cbf equal to 1, cu_sbt_flag may be signaled to indicate whether the whole residual block or a sub-part of the residual block is coded. In the former case, inter MTS information is further parsed to determine the transform type of the CU. In the latter case, a part of the residual block is coded with inferred adaptive transform and the other part of the residual block is zeroed out.

[0277] When SBT is used for an inter-coded CU, SBT type and SBT position information are signaled in the bitstream. There are two SBT types and two SBT positions,

as indicated in Fig. 36. For SBT-V (or SBT-H), the TU width (or height) may equal to half of the CU width (or height) or 1/4 of the CU width (or height), resulting in 2:2 split or 1:3/3:1 split. The 2:2 split is like a binary tree (BT) split while the 1:3/3:1 split is like an asymmetric binary tree (ABT) split. In ABT splitting, only the small region contains the non-zero residual. If one dimension of a CU is 8 in luma samples, the 1:3/3:1 split along that dimension is disallowed. There are at most 8 SBT modes for a CU.

[0278] Position-dependent transform core selection is applied on luma transform blocks in SBT-V and SBT-H (chroma TB always using DCT-2). The two positions of SBT-H and SBT-V are associated with different core transforms. More specifically, the horizontal and vertical transforms for each SBT position is specified in Fig. 36. For example, the horizontal and vertical transforms for SBT-V position 0 is DCT-8 and DST-7, respectively. When one side of the residual TU is greater than 32, the transform for both dimensions is set as DCT-2. Therefore, the subblock transform jointly specifies the TU tiling, cbf, and horizontal and vertical core transform type of a residual block.

[0279] The SBT is not applied to the CU coded with combined inter-intra mode.

2.26. Template matching based adaptive merge candidate reorder

[0280] To improve the coding efficiency, after the merge candidate list is constructed, the order of each merge candidate is adjusted according to the template matching cost. The merge candidates are arranged in the list in accordance with the template matching cost of ascending order. It is operated in the form of sub-group.

[0281] The template matching cost is measured by the SAD (Sum of absolute differences) between the neighbouring samples of the current CU and their corresponding reference samples. If a merge candidate includes bi-predictive motion information, the corresponding reference samples are the average of the corresponding reference samples in reference list0 and the corresponding reference samples in reference list1, as illustrated in Fig. 37. If a merge candidate includes sub-CU level motion information, the corresponding reference samples consist of the neighbouring samples of the corresponding reference sub-blocks, as illustrated in Fig. 38.

[0282] The sorting process is operated in the form of sub-group, as illustrated in Fig. 39. The first three merge candidates are sorted together. The following three merge candidates are sorted together.

[0283] The template size (width of the left template or height of the above template) is 1. The sub-group size is 3.

2.27. Adaptive Merge Candidate List

[0284] It can assume the number of the merge candidates is 8. It takes the first 5 merge candidates as a first subgroup and take the following 3 merge candidates as a second subgroup (i.e., the last subgroup).

[0285] For the encoder, after the merge candidate list is constructed, some merge candidates are adaptively reordered in an ascending order of costs of merge candidates as shown in Fig. 40.

[0286] More specifically, the template matching costs for the merge candidates in all subgroups except the last subgroup are computed; then reorder the merge candidates in their own subgroups except the last subgroup; finally, the final merge candidate list will be got.

[0287] For the decoder, after the merge candidate list is constructed, some/no merge candidates are adaptively reordered in ascending order of costs of merge candidates as shown in Fig. 41. In Fig. 41, the subgroup the selected (signaled) merge candidate located in is called the selected subgroup.

[0288] More specifically, if the selected merge candidate is located in the last subgroup, the merge candidate list construction process is terminated after the selected merge candidate is derived, no reorder is performed and the merge candidate list is not changed; otherwise, the execution process is as follows:

[0289] The merge candidate list construction process is terminated after all the merge candidates in the selected subgroup are derived; compute the template matching costs for the merge candidates in the selected subgroup; reorder the merge candidates in the selected subgroup; finally, a new merge candidate list will be got.

[0290] For both encoder and decoder, a template matching cost is derived as a function of T and RT, wherein T is a set of samples in the template and RT is a set of reference samples for the template.

[0291] When deriving the reference samples of the template for a merge candidate, the motion vectors of the merge candidate are rounded to the integer pixel accuracy.

[0292] The reference samples of the template (RT) for bi-directional prediction are derived by weighted averaging of the reference samples of the template in reference list0 (RT_0) and the reference samples of the template in reference list1 (RT_1) as follows.

$$RT = \big((8 - w) * RT_0 + w * RT_1 + 4\big) \gg 3 \quad (2\text{-}33)$$

where the weight of the reference template in reference list0 (8-w) and the weight of the reference template in reference list1 (w) are decided by the BCW index of the merge candidate. BCW index equal to {0,1,2,3,4} corresponds to w equal to {-2,3,4,5,10}, respectively.

[0293] If the Local Illumination Compensation (LIC) flag of the merge candidate is true, the reference samples of the template are derived with LIC method.

[0294] The template matching cost is calculated based on the sum of absolute differences (SAD) of T and RT.

[0295] The template size is 1. That means the width of the left template and/or the height of the above template is 1.

[0296] If the coding mode is MMVD, the merge candidates to derive the base merge candidates are not reordered.

[0297] If the coding mode is GPM, the merge candidates to derive the uni-prediction candidate list are not reordered.

2.28. Geometric prediction mode with Motion Vector Difference

[0298] In Geometric prediction mode with Motion Vector Difference (GMVD), each geometric partition in GPM can decide to use GMVD or not. If GMVD is chosen for a geometric region, the MV of the region is calculated as a sum of the MV of a merge candidate and an MVD. All other processing is kept the same as in GPM.

[0299] With GMVD, an MVD is signaled as a pair of direction and distance. There are nine candidate distances (¼-pel, ½-pel, 1-pel, 2-pel, 3-pel, 4-pel, 6-pel, 8-pel, 16-pel), and eight candidate directions (four horizontal/vertical directions and four diagonal directions) involved. In addition, when pic_fpel_mmvd_enabled_flag is equal to 1, the MVD in GMVD is also left shifted by 2 as in MMVD.

2.29. Affine MMVD

[0300] In affine MMVD, an affine merge candidate (which is called, base affine merge

candidate) is selected, the MVs of the control points are further refined by the signalled MVD information.

[0301] The MVD information for the MVs of all the control points are the same in one prediction direction.

[0302] When the starting MVs is bi-prediction MVs with the two MVs point to the different sides of the current picture (i.e. the POC of one reference is larger than the POC of the current picture, and the POC of the other reference is smaller than the POC of the current picture), the MV offset added to the list0 MV component of starting MV and the MV offset for the list1 MV has opposite value; otherwise, when the starting MVs is bi-prediction MVs with both lists point to the same side of the current picture (i.e. POCs of two references are both larger than the POC of the current picture, or are both smaller than the POC of the current picture), the MV offset added to the list0 MV component of starting MV and the MV offset for the list1 MV are the same.

2.30. Adaptive decoder side motion vector refinement (ADMVR)

[0303] In ECM-2.0, a multi-pass decoder-side motion vector refinement (DMVR) method is applied in regular merge mode if the selected merge candidate meets the DMVR conditions. In the first pass, bilateral matching (BM) is applied to the 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).

[0304] Adaptive decoder side motion vector refinement method consists of the two new merge modes introduced to refine MV only in one direction, either L0 or L1, of the bi prediction for the merge candidates that meet the DMVR conditions. The multi-pass DMVR process is applied for the selected merge candidate to refine the motion vectors, however either MVD0 or MVD1 is set to zero in the 1st pass (i.e., PU level) DMVR.

[0305] Like the regular merge mode, merge candidates for the proposed merge modes are derived from the spatial neighboring coded blocks, TMVPs, non-adjacent blocks, HMVPs, and pair-wise candidate. The difference is that only those meet DMVR conditions are added into the candidate list. The same merge candidate list (i.e., ADMVR merge list) is used by the two proposed merge modes and merge index is coded as in regular merge mode.

2.31. IBC with Template Matching

[0306] It is proposed to also use Template Matching with IBC for both IBC merge mode and IBC AMVP mode.

[0307] The IBC-TM merge list has been modified compared to the one used by regular IBC merge mode such that the candidates are selected according to a pruning method with a motion distance between the candidates as in the regular TM merge mode. The ending zero motion fulfillment (which is a nonsense regarding Intra coding) has been replaced by motion vectors to the left (-W, 0), top (0, -H) and top-left (-W, -H) CUs, then, if necessary, the list is fulfilled with the left one without pruning.

[0308] In the IBC-TM merge mode, the selected candidates are refined with the Template Matching method prior to the RDO or decoding process. The IBC-TM merge mode has been put in competition with the regular IBC merge mode and a TM-merge flag is signaled.

[0309] In the IBC-TM AMVP mode, up to 3 candidates are selected from the IBC merge list. Each of those 3 selected candidates are refined using the Template Matching method and sorted according to their resulting Template Matching cost. Only the 2 first ones are then considered in the motion estimation process as usual.

[0310] The Template Matching refinement for both IBC-TM merge and AMVP modes is quite simple since IBC motion vectors are constrained to be integer and within a reference region as shown in Fig. 42. So, in IBC-TM merge mode, all refinements are performed at integer precision, and in IBC-TM AMVP mode, they are performed either at integer or 4-pel precision. In both cases, the refined motion vectors in each refinement step must respect the constraint of the reference region.

2.32. IBC Merge Mode with Block Vector Differences

[0311] IBC merge mode with block vector differences is shown as follows.

[0312] The distance set is {1-pel, 2-pel, 4-pel, 8-pel, 12-pel, 16-pel, 24-pel, 32-pel, 40-pel, 48-pel, 56-pel, 64-pel, 72-pel, 80-pel, 88-pel, 96-pel, 104-pel, 112-pel, 120-pel, 128-pel}, and the BVD directions are two horizontal and two vertical directions.

[0313] The base candidates are selected from the first five candidates in the reordered IBC merge list. And based on the SAD cost between the template (one row above and one column left to the current block) and its reference for each refinement position, all the

possible MBVD refinement positions (20×4) for each base candidate are reordered. Finally, the top 8 refinement positions with the lowest template SAD costs are kept as available positions, consequently for MBVD index coding.

2.33. Reconstruction-Reordered IBC (RR-IBC)

[0314] Screen content coding tools like Intra Block Copy (IBC) generate a prediction block by directly copying a prior coded reference region in the same picture. Symmetry is often observed in video content, especially in text character regions and computer-generated graphics in screen content sequences, as shown in Fig. 43. Therefore, a specific screen content coding tool considering the symmetry would be efficient to compress such kinds of video contents.

[0315] A Reconstruction-Reordered IBC (RR-IBC) mode is proposed for screen content video coding. When it is applied, the samples in a reconstruction block are flipped according to a flip type of the current block. At the encoder side, the original block is flipped before motion search and residual calculation, while the prediction block is derived without flipping. At the decoder side, the reconstruction block is flipped back to restore the original block.

[0316] Two flip methods, horizontal flip and vertical flip, are supported for RR-IBC coded blocks. A syntax flag is firstly signalled for an IBC AMVP coded block, indicating whether the reconstruction is flipped, and if it is flipped, another flag is further signaled specifying the flip type. For IBC merge, the flip type is inherited from neighbouring blocks, without syntax signalling. Considering the horizontal or vertical symmetry, the current block and the reference block are normally aligned horizontally or vertically. Therefore, when a horizontal flip is applied, the vertical component of the BV is not signaled and inferred to be equal to 0. Similarly, the horizontal component of the BV is not signaled and inferred to be equal to 0 when a vertical flip is applied.

[0317] To better utilize the symmetry property, a flip-aware BV adjustment approach is applied to refine the block vector candidate. For example, as shown in Fig. 44, (x_{nbr}, y_{nbr}) and (x_{cur}, y_{cur}) represent the coordinates of the center sample of the neighboring block and the current block, respectively, BV^{nbr} and BV^{cur} denotes the BV of the neighboring block and the current block, respectively. Instead of directly inheriting the BV from a neighbouring block, the horizontal component of BV^{cur} is calculated by adding a motion shift to the horizontal component of BV^{nbr} (denoted as $BV^{nbr}{}_h$) in case that the neighbouring

block is coded with a horizontal flip, i.e., $BV^{cur}_h = 2(x_{nbr} - x_{cur}) + BV^{nbr}_h$. Similarly, the vertical component of BV^{cur} is calculated by adding a motion shift to the vertical component of BV^{nbr} (denoted as BV^{nbr}_v) in case that the neighbouring block is coded with a vertical flip, i.e., $BV^{cur}_v = 2(y_{nbr} - y_{cur}) + BV^{nbr}_v$.

2.34. Intra template matching

[0318] Intra template matching prediction (Intra TMP) is a special intra prediction mode that copies the best prediction block from the reconstructed part of the current frame, whose L-shaped template matches the current template. For a predefined search range, the encoder searches for the most similar template to the current template in a reconstructed part of the current frame and uses the corresponding block as a prediction block. The encoder then signals the usage of this mode, and the same prediction operation is performed at the decoder side.

[0319] The prediction signal is generated by matching the L-shaped causal neighbor of the current block with another block in a predefined search area in Fig. 45 consisting of:

R1: current CTU.
R2: top-left CTU.
R3: above CTU.
R4: left CTU.

[0320] SAD is used as a cost function.

[0321] Within each region, the decoder searches for the template that has least SAD with respect to the current one and uses its corresponding block as a prediction block.

[0322] The dimensions of all regions (SearchRange_w, SearchRange_h) are set proportional to the block dimension (BlkW, BlkH) to have a fixed number of SAD comparisons per pixel. That is:

SearchRange_w = a * BlkW.
SearchRange_h = a * BlkH.

where 'a' is a constant that controls the gain/complexity trade-off. In practice, 'a'is equal to 5.

[0323] The Intra template matching tool is enabled for CUs with size less than or equal to 64 in width and height. This maximum CU size for Intra template matching is configurable.

[0324] The Intra template matching prediction mode is signaled at CU level through a dedicated flag when DIMD is not used for current CU.

2.35. Non-local illumination compensation

[0325] One non-local illumination compensation (NL-IC) scheme is proposed. With the method, instead of using the template samples, the samples of the previously coded CUs are utilized for deriving the linear model used for the motion compensation of the current block. Specifically, after the reconstruction of each inter CU (except for GPM and SbTMVP CUs), one linear model is derived by minimizing the difference between the reconstruction and prediction samples of the block. Then, for both regular and subblock merge modes, up to 6 NL-IC candidates which are obtained from spatial adjacent and non-adjacent neighbors to current CU are inserted and reordered together with the existing candidates in the merge list. The fist N candidates with the smallest SADs remain in the list with one index being signaled to indicate which candidate is selected. In the contribution, N is kept the same as ECM-10.0, i.e., 10 for regular merge mode and 15 for subblock merge mode. Additionally, the same pattern used for obtaining spatial non-adjacent neighbors in regular merge is reused to locate the corresponding non-adjacent NL-IC candidates. When one IC-ILM candidate is selected, the associated linear model is used together with its motion information (e.g., MVs, CPMVs, reference picture indices and so forth) to generate the prediction samples of the CU.

2.36. LIC flag derivation for merge candidates with template costs

[0326] It is proposed to derive the LIC flag of a merge candidate based on template costs.

[0327] The LIC flag of a merge candidate is derived by comparing two template costs: a SAD-based template cost, denoted as C0, and a Mean Removal SAD (MRSAD)-based template cost, denoted as C1. The LIC flag is set to be false, if C0 <= C1 and is set to be true, if C0 > C1.

To favor the inherited LIC flag, C0 is multiplied by α if the inherited LIC flag is false while C1 is multiplied by α if the inherited LIC flag is true, where $\alpha < 1$.

3. Problems

[0328] In current design of LIC, the LIC parameters are derived using the reconstructed

samples of nearest row and/or column. These derived parameters may be not optimal for all blocks, the coding performance can be improved by adjusting the parameters.

4. Detailed Solutions

[0329] The detailed solutions 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.

[0330] In the present disclosure, Local illumination compensation (LIC) may not be limited to the current LIC technology. LIC may refer to an inter prediction technique to model local illumination variation between current block and its prediction block as a function of that between current block template and reference block template. The parameters of the function may be denoted by a linear equation (e.g., α*p[x]+β) or a non-linear equation.

LIC with slope adjustment

1. It is proposed that a function f(p[x, y]) is used to derive a refined prediction sample of a prediction sample in a video unit, and the one or more parameters of the function may be modified. Denote the coding mode as LIC_SLOPE.
    a. In one example, f is any function, such as a linear model used in LIC.
    b. In one example, the function may be a linear equation.
        i. In one example, f(p[x, y]) = α * p[x, y] + β, wherein α and β denotes the parameters of the linear equation.
    c. In one example, more than 2 parameters may be used in the function.
        i. In one example, the adjacent and/or non-adjacent neighbouring prediction samples of p[x, y] may be used.
            1) In one example, f(p[x, y]) = a_0 * p[x, y] + a_1*p[x–1, y] + a_2*p[x+1, y] + a_3*p[x1, y–1] + a_4*p[x, y+1] + b.
                a) In one example, a_1, or a_2, or a_3, or a_4 may be 0.
    d. In one example, the function may be a non-linear equation.
        i. In one example, f(p[x, y]) = α_0 * p[x, y] + α_1 * p[x, y] * p[x, y]+ β.
        ii. In one example, f(p[x, y]) = a_0 * p[x, y] + a_1*p[x–1, y] + a_2*p[x+1, y] + a_3*p[x1, y–1] + a_4*p[x, y+1] + a_5 * (p[x, y]) * (p[x, y]) + b.
    e. In one example, a method may be applied to derive the parameters of the function using a set of training samples (which may be reconstructed samples) by minimizing the difference between the ground truth (G) and the prediction (P) of the training samples.
        i. In one example, the difference may refer to the absolute transformed difference (SATD), the sum of the squared errors (SSE), or the sum of the

absolute difference (SAD), or the mean removal sum of the absolute difference (MRSAD), or a subjective quality metric (e.g., the structural similarity index measure (SSIM)).

ii. In one example, G may refer to the reconstruction of a template and P may refer to one or more prediction signal of the template.

1) In one example, the template may consist of the neighbouring reconstructed samples of the video unit.

a) In one example, the neighbouring reconstructed samples may be adjacent and/or non-adjacent.

2) In one example, the prediction signal of the template may be derived using the same method that is used to derive the prediction of the video unit.

a) Alternatively, one or more coding information may be different from the video unit.

i. In one example, one or more coding tools may be not applied when deriving the prediction of the template, but they are applied when deriving the prediction of the video unit.

iii. In one example, G may refer to the reconstruction of a video unit and P may refer to one or more prediction signal of the video unit.

iv. In one example, G may refer to the prediction/reconstruction of luma video unit and P may refer to the prediction/reconstruction of chroma video unit.

v. In one example, the Least-Mean-Square (LMS) method may be used.

vi. In one example, the LDL method may be used.

vii. In one example, the Gaussian elimination may be used, e.g., the method used in CCCM.

f. In one example, one or more parameters of function (e.g., α and β) may be modified using adjustment parameters.

i. In one example, an adjustment parameter may be used to adjust α, such as $\alpha + u$ or $\alpha \times u$.

ii. In one example, an adjustment parameter may be used to adjust β, such as $\beta + v$ or $\beta \times v$.

iii. In one example, an adjustment parameter may be used to adjust more than one parameters, such as α and β.

1) In one example, $\alpha' = \alpha + u$ and $\beta' = \beta - u * m$, where u is the adjustment parameter and m is a variable which is derived using the samples that are used to derive the parameters of the function.

a) In one example, m may be calculated depending on the samples that are used to derive the LIC parameters.

i. In one example may be equal to the mean of the samples.

2) In one example, $\alpha' = \big((\alpha \ll S) + u + offset\big) \gg S$, where S and *offset* are integers.

a) E.g., $offset = 1 \ll (S - 1)$.

3) In one example, the unit of the slope adjustment parameter may be $1/N^{th}$ of a chroma sample value per one luma sample value, such as N = 2, or 4, or 8, or 16, or 32.
4) In one example, the slope adjustment parameter may be an integer not equal to zero.
a) In one example, the slope adjustment parameter may be between T_1 and T_2, inclusive.
i. In one example, $T_1 = -2$ and $T_2 = 2$, or $T_1 = -4$ and $T_2 = 4$, or $T_1 = -8$ and $T_2 = 8$, or $T_1 = -16$ and $T_2 = 16$.
ii. In one example, the absolute value of T_1 may be not equal to the absolute value of T_2.

iv. In one example, the slope adjustment parameter and/or the indication of the slope adjustment parameter may be signalled in the bitstream.
1) In one example, the sign and the absolute value of the slope adjustment parameter may be signalled using one or more syntax elements.
a) In one example, the syntax elements may be binarized with fixed length coding, or truncated unary coding, or unary coding, or EG coding, or coded a flag.
b) In one example, the syntax element may be bypass coded or context coded.
c) In one example, a pre-defined absolute value is used and is not signalled.
i. In one example, the absolute value may be equal to 1/2/3/4.
2) In one example, a set of adjustment parameters may be pre-defined/derived/signalled, and an indication is signalled in the bitstream.

v. In one example, the adjustment parameters and/or the indication of the adjustment parameters may be derived using coding information.
vi. In one example, a reordering method may be applied when signalling and/or deriving the slope adjustment parameter.
1) In one example, a template of the video unit consists of the neighbouring reconstructed samples may be used in the reordering.
a) In one example, the i-th prediction template (TP_i) is obtained using the i-th slope adjustment parameter (S_i) and a cost (C_i) is calculated between TP_i and the i-th template, and partial or all the slope adjustment parameters are reordered according to the costs.
b) In one example, the slope adjustment parameter with the lowest cost may be used to modify the parameters of the function.
c) In one example, the first N best slope adjustment parameters may be signalled.
i. In one example, the index of the first N best slope adjustment parameters may be signalled.
d) In one example, the template size may be a constant value (e.g., 1, or 2, or 4) or an adaptive value depending on block dimensions/size.

vii. In one example, the adjustment parameters and/or the indication of the adjustment parameters may be inherited.
1) In one example, the indication of the adjustment parameter of the video unit may be inherited from the neighbouring video units (adjacent or non-adjacent).
viii. In one example, the number of adjustment parameters (Nap) may be predefined/signalled/derived, wherein Nap is an integer larger than 0.
1) In one example, Nap = 1, or 2, or 3, or 4, or 6, or 8.

2. In one example, LIC_SLOPE may be applied to a specific prediction mode, such as intra, or inter, or IBC, or Palette, or other prediction modes.
3. In one example, LIC_SLOPE may be applied to a specific coding tool.
a. In one example, the coding tool may refer to inter coding tool.
i. In one example, inter coding tool may refer to a specific coding tool such as CIIP (e.g., CIIP-Planar, CIIP-TIMD, CIIP-TM), BCW (e.g., BCW index derived by TM), MMVD (e.g., MMVD or TM based reordering for MMVD), template matching (TM), affine (e.g., affine-MMVD, TM based reordering for affine MMVD), DMVR/multi-pass DMVR, PROF, BDOF/sample based BDOF, adaptive decoder-side motion vector refinement (ADMVR), OBMC or TM based OBMC, MHP, GPM (e.g., GPM, GPM-TM, GPM-MMVD, GPM-intra, GPM-Affine, GPM-IBC, regression-based GPM blending), bilateral/template matching AMVP-merge mode, InterCCCM, Inter CCP merge mode, RPR, or their variants.
ii. Alternatively, LIC_SLOPE may be not used for one or more above inter coding tools.
1) In one example, LIC_SLOPE may be not used for Affine mode.
2) In one example, LIC_SLOPE may be not used for AMVP-merge mode.
3) In one example, LIC_SLOPE may be not used for BCW.
4) In one example, LIC_SLOPE may be not used for AMVR.
a) Alternatively, LIC_SLOPE may be not used for a specific motion vector resolution.
5) In one example, LIC_SLOPE may be not used for SMVD.
6) In one example, LIC_SLOPE may be not used for RPR.
7) In one example, LIC_SLOPE may be not used in the reordering process of motion vectors or motion candidates.
8) In one example, LIC_SLOPE may be not used in the template matching, or a method which uses template matching to obtain the prediction.
9) In one example, when BCW, or AMVR, or SMVD is used, the syntax elements indicating whether LIC_SLOPE is used are not signalled.
10) In one example, LIC_SLOPE may be not used for a cross-component coding tool which uses the cross-component information to get the prediction/reconstruction.
a) In one example, the cross-component coding tool may refer to CCLM, and/or MMLM, and/or GLM, and/or CCCM, and/or interCCCM, and/or CCP merge, and/or inter CCP merge, and/or their variants.

b) In one example, when LIC_SLOPE is used, the syntax elements indicating whether the cross-component coding tool is used are not signalled.
c) In one example, the determination whether to use LIC_SLOPE with the cross-component coding tool may depend on information.
i. In one example, the information may refer to block dimensions/size.
ii. In one example, the information may refer to the picture/slice type.
1. In one example, when the picture/slice is a low-delay picture/slice, LIC_SLOPE is disallowed to be used with the cross-component coding tool.
d) Alternatively, LIC_SLOPE may to be used with the cross-component coding tool.
i. In one example, when LIC_SLOPE is used for the cross-component coding tool, the parameters of LIC for luma and/or chroma components may be not adjusted.
ii. Alternatively, when LIC_SLOPE is used for the cross-component coding tool, LIC may be disabled for luma and/or chroma components.
iii. In one example, a pre-defined/signaled/derived adjustment parameter may be used for the block based reference picture derivation.
1) In one example, a pre-defined adjustment parameter (Delta) may be used for the block based reference picture derivation, such as Delta = 1/–1/2/–2/3/–3.
2) In one example, LIC_SLOPE is not used during the block based reference picture derivation.
b. In one example, the coding tool may refer to BDPCM or Palette.
c. In one example, the coding tool may refer to IBC coding tool.
i. In one example, IBC coding tool refer to IBC AMVP mode may refer to normal IBC AMVP, or TM based IBC AMVP, or RR-IBC AMVP mode, or other IBC AMVP mode wherein a BV predictor is derived and BVD is signalled/derived, or, IBC merge mode may refer to normal IBC merge mode, or IBC-TM merge mode, or IBC-MBVD mode, or IBC-CIIP, or IBC-GPM, or IBC-LIC, or sub-pel based IBC, filtering based IBC, or their variants.
d. In one example, LIC_SLOPE may be applied to uni-prediction and/or bi-prediction.
i. In one example, when LIC_SLOPE is applied to bi-prediction, one slope adjustment parameter may be used.
ii. Alternatively, when LIC_SLOPE is applied to bi-prediction, two slope adjustment parameters may be used, where each slope adjustment is used to modify the LIC parameters for one direction (L0 or L1).
iii. Alternatively, LIC_SLOPE is not allowed to be used for bi-prediction.

e. In one example, when LIC_SLOPE is used, one or more above coding tool may be not used.
    i. In one example, OBMC is not used.
        1) In one example, the indication of OBMC may be not signalled when LIC_SLOPE is used.

4. In one example, LIC_SLOPE may be applied to AMVP mode.
    a. The AMVP mode may be regular AMVP mode, affine AMVP mode, TM AMVP mode, or IBC AMVP mode.
    b. In one example, a slope adjustment parameter or an indication of the slope adjustment parameter may be signalled for AMVP mode, which is used to modify the LIC parameters.
    c. In one example, LIC_SLOPE may be not used for TM AMVP mode.
        i. In one example, LIC_SLOPE may be not used to construct the TM based AMVP candidate list.
    d. In one example, LIC_SLOPE may be used for TM AMVP mode but not used to construct the TM based AMVP candidate list.
5. In one example, LIC_SLOPE may be applied to merge mode.
    a. The merge mode may be regular merge mode, sub-block-based merge mode, TM merge mode, affine merge mode, MMVD, or IBC merge mode, or GPM, or other merge modes.
    b. In one example, the slope adjustment parameter or the indication of the slope adjustment parameter, or whether LIC_SLOPE is used may be inherited for merge mode, which is used to modify the LIC parameters.
    c. Alternatively, the slope adjustment parameter or the indication of the slope adjustment parameter, or whether LIC_SLOPE is used may be signalled for merge mode.
    d. Alternatively, the slope adjustment parameter or the indication of the slope adjustment parameter, or whether LIC_SLOPE is used may be derived for merge mode.
    e. In one example, LIC_SLOPE may be applied to ARMC process.
        i. In one example, LIC_SLOPE may be applied to ARMC process of a specific coding tool.
        ii. Alternatively, LIC_SLOPE may be not applied to ARMC process.
    f. In one example, the LIC parameters after being modified by the slope adjustment parameter may be used with other LIC parameters (e.g., inherited LIC parameters).
        i. In one example, a list of LIC parameters is constructed and the best LIC parameter may be derived or signalled.
            1) In one example, reordering may be used for the list.
            2) In one example, pruning may be used for the list.
            3) In one example, similar check may be used for the list, where a set LIC parameters is not added to the list when it is similar to one of the existing LIC parameters in the list.
    g. In one example, reordering may be used for the slope adjustment parameters.

i. In one example, a template consisting of neighbouring samples may be used for the reordering, where the different LIC parameters obtained using different slope adjustment parameters are used for the template and template cost is calculated and sort.
ii. In one example, the template size may be N_S, such as N_S = 1/2/3/4/5/6/7/8 or an adaptive integer.

6. In one example, the slope adjustment parameters and/or LIC parameters of the current video unit may be used by the following video units.
   a. In one example, the LIC parameters of the current video unit may refer to those parameters used to refine the prediction for the current video unit.
   b. In one example, the LIC parameters of the current video unit may be derived using the template.
   c. In one example, the LIC parameters of the current video unit may be derived using the prediction and reconstruction of the current video unit.
   d. In one example, the slope adjustment parameters and/or LIC parameters may be stored in a history table.
   e. In one example, the slope adjustment parameters and/or LIC parameters may be used by the following spatial adjacent and/or non-adjacent neighbouring video units.
   f. In one example, the slope adjustment parameters and/or LIC parameters may be used by the following video units in the same CTU/CTU row/slice/tile/picture, and/or video units in a different CTU/CTU row/slice/tile/picture.
   g. In one example, the slope adjustment parameters and/or LIC parameters of a luma video unit may be used for a chroma video unit.
   h. In one example, one or more parameters may be modified when the slope adjustment parameters and/or LIC parameters are used for the following video units.
7. Whether to and/or how to apply LIC_SLOPE may depend on coding information, the coding information may refer to:
   a. whether a specific coding method is allowed, such as LIC, LIC with multiple templates, LIC with multiple models, AMVP-merge mode
   b. block dimensions and/or block size
      i. In one example, the block is disallowed to be coded with LIC_SLOPE when the block size (W×H) is larger than or equal to a threshold (T1), wherein W and H denotes block width and block height, respectively.
         1) In one example, T1 = 64, or 128, or 256, or 512, or 1024, or 2048, or 4096, or 8192.
      ii. In one example, the block is disallowed to be coded with LIC_SLOPE when the block size (W×H) is less than or equal to a threshold (T2), wherein W and H denotes block width and block height, respectively.
         1) In one example, T2 = 16, or 32, or 64, or 128, or 256.
      iii. In one example, the block is disallowed to be coded with LIC_SLOPE when W is larger than or equal to a threshold (T3) and/or H is larger than or equal to a threshold (T4).
         1) In one example, T3 = 32/64/128/256.
         2) In one example, T4 = 32/64/128/256.

iv. In one example, the block is disallowed to be coded with LIC_SLOPE when W is less than or equal to a threshold (T5) and/or H is less than or equal to a threshold (T6).
1) In one example, T5 = 4/8/16/32/64.
2) In one example, T6 = 4/8/16/32/64.
v. In one example, the block is disallowed to be coded with LIC_SLOPE when W/H and/or H/W is larger than or equal to a threshold (T7).
1) In one example, T7 = 2/4/8/16/32.
vi. In one example, the above thresholds (e.g., T1/T2/T3/T4/T5/T6/T7) may depend on slice/picture type.
vii. In one example, the block size may refer to luma block size.
viii. In one example, the block size may refer to chroma block size.
c. block depth
d. slice/picture type and/or partition tree type (single, or dual tree, or local dual tree)
e. temporal layer identification
i. In one example, the block is disallowed to be coded with LIC_SLOPE when the temporal layer identification is less than or equal to TidTh1.
1) In one example, TidTh1 = 0/1/2.
ii. In one example, the block is disallowed to be coded with LIC_SLOPE when the temporal layer identification is larger than or equal to TidTh2.
1) In one example, TidTh2 = 3/4/5/6.
f. block location
i. In one example, when the block locates at the left-top boundary of the picture/slice/tile, it is disallowed to use LIC_SLOPE for the block.
g. colour format
h. colour component
i. In one example, LIC_SLOPE may apply to all colour components.
ii. In one example, when LIC_SLOPE is applied to chroma components, it may be different from that for luma component.
1) In one example, the slope adjustment parameter may be different.
a) In one example, whether to and/how to derive/signal the slope adjustment parameter may be different.
iii. In one example, whether to and/or how to apply LIC_SLOPE to a first component may depend on whether to and/or how to apply LIC_SLOPE to a second component.
1) In one example, the first component may refer to chroma component (e.g., Cb and/or Cr), and the second component may refer to luma component (e.g., Y).
2) In one example, the way to apply LIC_SLOPE to the first component may be same as the second component.
a) Alternatively, the way to apply LIC_SLOPE to the first component may be different from the second component.
3) In one example, the slope adjustment parameters and/or the indication of the slope adjustment parameters of the current chroma block may be inherited from luma block.
iv. In one example, LIC_SLOPE may be applied to luma component, but not to chroma components.

1) In one example, luma component may refer to Y in YCbCr colour space or G in RGB colour space.
2) In one example, chroma components may refer to Cb and/or Cr in YcbCr colour space or R and/or B in RGB colour space.

8. Indication of LIC_SLOPE may be conditionally signalled wherein the condition may include:
   a. whether LIC is allowed/used
   b. whether AMVP mode is used
   c. whether uni-prediction/bi-prediction is used
      i. In one example, the indication of the LIC_SLOPE may be not signalled when bi-prediction is used.
   d. whether AMVP-merge mode is allowed/used
   e. whether LIC with multiple templates is allowed/used
   f. whether LIC with multiple models is allowed/used
   g. block dimensions and/or block size
      i. In one example, the indication of the LIC_SLOPE may be not signalled when the block size (W×H) is larger than or equal to a threshold (N1), wherein W and H denotes block width and block height, respectively.
         1) In one example, N1 = 64, or 128, or 256, or 512, or 1024, or 2048, or 4096.
      ii. In one example, the indication of the LIC_SLOPE may be not signalled when the block size (W×H) is less than or equal to a threshold (N2), wherein W and H denotes block width and block height, respectively.
         1) In one example, N2 = 16, or 32, or 64, or 128, or 256.
      iii. In one example, the indication of LIC_SLOPE may be not signalled when W is larger than or equal to a threshold (N3) and/or H is larger than or equal to a threshold (N4).
         1) In one example, N3 = 32/64/128/256.
         2) In one example, N4 = 32/64/128/256.
      iv. In one example, the indication of LIC_SLOPE may be not signalled when W is less than or equal to a threshold (N5) and/or H is less than or equal to a threshold (N6).
         1) In one example, N5 = 4/8/16/32/64.
         2) In one example, N6 = 4/8/16/32/64.
      v. In one example, the indication of LIC_SLOPE may be not signalled when W/H and/or H/W is larger than or equal to a threshold (N7).
         1) In one example, N7 = 2/4/8/16/32.
      vi. In one example, the above thresholds (e.g., N1/N2/N3/N4/N5/N6/N7) may depend on slice/picture type.
      vii. In one example, the block size may refer to luma block size.
      viii. In one example, the block size may refer to chroma block size.
   h. block depth
   i. slice/picture type and/or partition tree type (single, or dual tree, or local dual tree)
   j. temporal layer identification
      i. In one example, the indication of LIC_SLOPE may be not signalled when the temporal layer identification (TID) is less than or equal to TidTh3.

1) In one example, TidTh3 = 0/1/2.

ii. In one example, the indication of LIC_SLOPE may be not signalled when the temporal layer identification (TID) is larger than or equal to TidTh4.

1) In one example, TidTh4 = 3/4/5/6.

k. block location

i. In one example, when the block locates at the left-top boundary of the picture/slice/tile, the indication of LIC_SLOPE may be not signalled.

l. colour format

m. colour component.

9. Whether current block is coded with LIC_SLOPE mode may be signalled using one or more syntax elements (SE).

a. In one example, the syntax element may be binarized with fixed length coding, or truncated unary coding, or unary coding, or EG coding, or coded a flag.

b. In one example, the syntax element may be bypass coded or context coded.

i. The context may depend on coded information, such as block dimensions, and/or block size, and/or slice/picture types, and/or the information of neighbouring blocks (adjacent or non-adjacent), and/or the information of other coding tools used for current block, and/or the information of temporal layer.

c. In one example, the one or more syntax elements may be signalled at sequence header/picture header/SPS/VPS/DPS/DCI/PPS/APS/slice header/tile group header.

d. In one example, the syntax element may be coded in a predictive way.

i. For example, the syntax element of the current block may be predicted by that of a neighboring block.

e. In one example, the syntax element may be coded conditionally.

i. For example, only if a first SE indicates that LIC_SLOPE is applicable, a second SE may be signaled to indicate whether LIC_SLOPE is used.

1) The first SE may be at sequence header/picture header/SPS/VPS/DPS/DCI/PPS/APS/slice header/tile group header.

a) In one example, the first SE may refer to an SPS/PPS/slice flag.

b) In one example, the first SE may refer to a general constraint information syntax for LIC_SLOPE and/or LIC.

2) The second SE may be for a block.

ii. For example, only if the second SE indicates LIC_SLOPE is used, a third SE may be signaled to indicate how to perform LIC_SLOPE.

10. It is proposed that the context of a SE of LIC (e.g., LIC flag) or SEs related to LIC may depend on coding information.

a. In one example, the coding information may refer to neighbouring video units.

i. In one example, the context may depend on the status of the SE of neighbouring video units.

1) In one example, a context ctxA may be used when LIC is not used for left video unit and LIC is not used for above video unit; a context ctxB may be used when LIC is used for left video unit but LIC is not used for above video unit, or when LIC is not used for left video unit

but LIC is used for above video unit; a context ctxC may be used when LIC is used for both left and above video units.
a) In one example, ctxA != ctxB != ctxC.
b) In one example, ctxA != ctxB = ctxC.

b. In one example, the coding information may refer to the dimensions and/or size of the current video unit.
c. In one example, the coding information may refer to slice/picture type and/or partition tree type (single, or dual tree, or local dual tree).
d. In one example, the coding information may refer to temporal layer identification.
e. In one example, LIC is used for a video unit may refer to a LIC flag is signaled for the video unit and/or a LIC flag is inherited for the video unit.
f. In one example, LIC is used for a video unit may refer to the video unit is coded with AMVP mode, and/or AMVP-merge mode, and/or merge mode.
g. In one example, the above methods may be applied when the LIC flag is signalled for merge mode.

11. In one example, the signalling of a SE of a coding tool may depend on whether LIC is used.
    a. In one example, whether to signal the SE of the coding tool may depend on whether LIC is used.
    b. In one example, the context of signalling the SE of the coding tool may depend on whether LIC is used.
    c. In one example, the coding tool may refer to an inter coding tool.
        i. In one example, the OBMC flag indicating whether OBMC is used may depend on whether LIC is used.
        ii. In one example, the context of signaling OBMC flag may depend on whether LIC is used.
            1) For example, a first context ctx1 may be used for OBMC flag when LIC is used, and a second context ctx2 may be used for OBMC flag when LIC is not used.
    d. In one example, the context of a SE of OBMC (e.g., OBMC flag) or SEs related to OBMC may depend on coding information.
        i. In one example, the coding information may refer to neighbouring video units.
        ii. In one example, the coding information may refer to the dimensions and/or size of the current video unit.
        iii. In one example, the coding information may refer to slice/picture type and/or partition tree type (single, or dual tree, or local dual tree).
        iv. In one example, the coding information may refer to temporal layer identification.
12. In one example, whether to and/or how to perform LIC flag derivation for merge candidates may depend on coding information.
    a. In one example, the coding information may refer to the template size/shape/dimensions.
    b. In one example, the coding information may refer to a value calculated using the template and/or the prediction of the template.
        i. In one example, the mean value of the template and/or the prediction of the template may be used.

c. In one example, the coding information may refer to whether the merge candidate is derived from a video unit coded with a specific coding tool.
    i. In one example, the coding tool may refer to LIC_SLOPE, or LIC with multiple templates, or LIC with multiple models, or other IC (illumination compensation) methods.
    ii. In one example, the derivation may be skipped when the merge candidate is derived from a video unit coded with the specific coding tool.

General aspects

13. 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.
14. 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.
15. Whether and/or how to apply the above methods may depend on the following information:
    a. A message signalled in the DPS/SPS/VPS/PPS/APS/picture header/slice header/tile group header/coding tree unit (CTU)/Coding unit (CU)/CTU row/group of CTUs/TU/PU block/Video coding unit
    b. Position of CU/PU/TU/block/Video coding unit
    c. Block dimension of current block and/or its neighbouring blocks
    d. Block shape of current block and/or its neighbouring blocks
    e. coded mode of a block, e.g., IBC or non-IBC inter mode or non-IBC subblock mode
    f. Indication of the colour format (such as 4:2:0, 4:4:4)
    g. Coding tree structure
    h. Slice/tile group type and/or picture type
    i. Colour component (e.g., may be only applied on chroma components or luma component)
    j. Temporal layer ID
    k. Profiles/Levels/Tiers of a standard
16. A syntax element disclosed above may be binarized as a flag, a fixed length code, an EG(x) code, a unary code, a truncated unary code, a truncated binary code, etc. It can be signed or unsigned.
17. A syntax element disclosed above may be coded with at least one context model. Or it may be bypass coded.
18. A syntax element disclosed above may be signaled in a conditional way.
    a. The SE is signaled only if the corresponding function is applicable.
    b. The SE is signaled only if the dimensions (width and/or height) of the block satisfy a condition.
19. A syntax element disclosed above may be signaled at block level/ sequence level/group of pictures level/picture level/slice level/tile group level, such as in coding structures of

CTU/CU/TU/PU/CTB/CB/TB/PB, or sequence header/picture header/SPS/VPS/DPS/DCI/PPS/APS/slice header/tile group header.

20. The proposed method(s) may be combined with another coding tool such as affine/MTS/LFNST/MMVD/MIP/ISP/CCLM/CCCM/SMVD/BDOF/DMVR/HMVP/Template Matching/IBC/Palette/etc.
21. The proposed method(s) may be excluded with another coding tool such as affine/MTS/LFNST/MMVD/MIP/ISP/CCLM/CCCM/SMVD/BDOF/DMVR/HMVP/Template Matching/IBC/Palette/etc.
    a. In one example, if the proposed method(s) is used, the excluded coding tool is disabled implicitly without signaling.
    b. In one example, if the excluded coding tool is used, the proposed method(s) is disabled implicitly without signaling.

[0331] As used herein, the term "video unit" or "video block" may be a sequence, a picture, a slice, a tile, a brick, a subpicture, a coding tree unit (CTU)/coding tree block (CTB), a CTU/CTB row, one or multiple coding units (CUs)/coding blocks (CBs), one or multiple CTUs/CTBs, one or multiple Virtual Pipeline Data Unit (VPDU), a sub-region within a picture/slice/tile/brick. The terms 'block' may represent a coding tree block (CTB), a coding tree unit (CTU), a coding block (CB), a CU, a PU, a TU, a PB, a TB.

[0332] The term "motion vector" or "block vector" may refer to a vector of horizontal and vertical displacements between the locations of a reference block and the current block. The reference block can be a video unit in a reference picture in the RPL list. The reference block can also be a video unit in the current picture.

[0333] Fig. 46 illustrates a flowchart of a method 4600 for video processing in accordance with embodiments of the present disclosure. The method 4600 is implemented during a conversion between a video unit of a video and a bitstream of the video.

[0334] At block 4610, for a conversion between a video unit of a video and a bitstream of the video, a refined prediction sample of a prediction sample in the video unit by is derived applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode. For example, the target coding mode may be an LIC-slope mode. It is noted that the target coding mode may be any suitable coding modes. One or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool. In some embodiments, the prediction mode comprises at least one of: an intra prediction mode, an inter prediction mode, an intra block copy (IBC) prediction mode, or other prediction mode.

[0335] At block 4620, the conversion is perfomred based on the refined prediction

sample. In some embodiments, the conversion includes encoding the video unit into the bitstream. In some other embodiments, the conversion includes decoding the video unit from the bitstream. In this way, the coding performance and coding efficiency can be improved by adjusting the parameters.

[0336] In some embodiments, if the target coding mode is used, one or more coding tools are not used. In some embodiments, overlapped block motion compensation (OBMC) is not used. In some embodiments, an indication of OBMC is not signaled, if the target coding mode is used. In some embodiments, the target coding mode is not allowed to be used for bi-prediction. In some embodiments, the target coding mode is not used for RPR. In some embodiments, if at least one of: BCW, adaptive motion vector resolution (AMVR), or symmetric motion vector differences (SMVD) is used, a syntax element indicating whether the target coding mode is used is not signaled.

[0337] In some embodiments, at least one of: pre-defined adjustment parameter, signaled adjustment parameter or derived adjustment parameter is used for a block based reference picture derivation. In some embodiments, the pre-defined adjustment parameter is used for the block based reference picture derivation. In some embodiments, the pre-defined adjustment parameter is equal to one of: 1, -1, 2, -2, 3, or -3. In some embodiments, the target coding mode is not used during a block based reference picture derivation.

[0338] In some embodiments, the target coding mode is applied to one or more inter coding tools, or wherein the target coding mode is not applied to one or more inter coding tools. In some embodiments, the one or more inter coding tools comprise at least one of: a combined inter and intra prediction (CIIP), a variant of CIIP, a bi-prediction with coding unit-level weight (BCW), a variant of BCW a merge mode with motion vector difference (MMVD), a variant of MMVD, a template matching (TM), a variant of TM, an affine, a variant of affine, a decoder side motion vector refinement (DMVR), a variant of DMVR, a multi-pass DMVR, a prediction refinement with optical flow (PROF), a variant of PROF, a bi-directional optical flow (BDOF), a variant of BDOF, a sample based BDOF, an adaptive decoder-side motion vector refinement (ADMVR), a variant of ADMVR, an OBMC, a variant of OBMC, a TM based OBMC, a multi-hypothesis prediction (MHP), a variant of MHP, a geometric partitioning mode (GPM), a variant of GPM, a bilateral matching advanced motion vector prediction (AMVP)-merge mode, a template matching AMVP-merge mode, an Inter convolutional cross-component model (CCCM), an Inter cross-component prediction (CCP) merge mode, a reference picture resampling (RPR), or

a variant of RPR. In some embodiments, the CIIP comprises at least one of: CIIP-Planar, CIIP- template-based intra mode derivation (TIMD), or CIIP-TM, or wherein the BCW comprises BCW index derived by TM, or wherein the MMVD comprises a TM based reordering for MMVD, or wherein the affine comprises at least one of: affine-MMVD, TM based reordering for affine MMVD, or wherein the GPM comprises at least one of: GPM-TM, GPM-MMVD, GPM-intra, GPM-Affine, GPM-IBC, regression-based GPM blending.

[0339] In some embodiments, the target coding mode is not used for a cross-component coding tool which uses cross-component information to get the prediction or reconstruction. In some embodiments, the cross-component coding tool comprises at least one of: a cross-component linear model (CCLM), a variant of CCLM, a multi-model CCLM (MMLM), a variant of MMLM, a gradient linear model (GLM), a variant of GLM, an inter convolutional cross-component model (CCCM), a variant of inter CCCM, a CCP merge, or a variant of CCP merge.

[0340] In some embodiments, the target coding mode is not used with the cross-component coding tool. In some embodiments, if the target coding mode is used for the cross-component coding tool, parameters of LIC for luma and/or chroma components are not adjusted. In some embodiments, if the target coding mode is used for the cross-component coding tool, LIC is disabled for luma and/or chroma components. In some embodiments, if the target coding mode is used, a syntax element indicating whether the cross-component coding tool is used is not signalled.

[0341] In some embodiments, a determination of whether to use the target coding mode with the cross-component coding tool depends on information. In some embodiments, the information comprises block dimensions or block size. Alternatively, or in addition, the information comprises picture type or slice type.

[0342] In some embodiments, if the picture type if a low-delay picture or the slice type is a low-delay slice, the target coding mode is disallowed to be used with the cross-component coding tool. In some embodiments, the target coding mode is not used for Affine mode. In some embodiments, the target coding mode is not used for AMVP-merge mode. In some embodiments, the target coding mode is not used for BCW.

[0343] In some embodiments, the target coding mode is not used for AMVR. Alternatively, the target coding mode is not used for a specific motion vector resolution.

[0344] In some embodiments, the target coding mode is not used for SMVD. In some embodiments, the target coding mode is not used in a reordering process of motion vectors or motion candidates.

[0345] In some embodiments, the target coding mode is not used in the template matching, or a method which uses template matching to obtain the prediction. In some embodiments, the coding tool comprises block-based delta pulse code modulation (BDPCM) or Palette. In some embodiments, the coding tool comprises an IBC coding tool. In some embodiments, the IBC coding tool comprises at least one of: an IBC AMVP mode, a variant of IBC AMVP, an IBC merge mode, a variant of IBC merge, an IBC-CIIP mode, a variant of IBC-CIIP, an IBC-GPM mode, a variant of IBC-GPM, an IBC-LIC mode, a variant of IBC-LIC, a sub-pel based IBC mode, a variant of sub-pel based IBC, a filtering based IBC mode, or a variant of filtering based IBC.

[0346] In some embodiments, the IBC AMVP mode comprises at least one of a normal IBC AMVP mode, a TM based IBC AMVP mode, or a reconstruction-reordered IBC (RR-IBC), other IBC AMVP mode where a block vector (BV) predictor is derived and block vector difference (BVD) is signalled or derived, or wherein the IBC merge mode comprises at least one of a normal IBC merge mode, an IBC-TM merge mode, or an IBC-merge mode with block vector differences (IBC-MBVD) mode.

[0347] In some embodiments, the target coding mode is applied to uni-prediction and/or bi-prediction. In some embodiments, if the target coding mode is applied to bi-prediction, one slope adjustment parameter is used. In some embodiments, if the target coding mode is applied to bi-prediction, two slope adjustment parameters are used, wherein each slope adjustment is used to modify LIC parameters for one direction.

[0348] In some embodiments, the target coding mode is applied to AMVP mode. Alternatively, or in addition, the target coding mode is applied to merge mode.

[0349] In some embodiments, the AMVP mode comprises at least one of: a regular AMVP mode, an affine AMVP mode, a TM AMVP mode, or an IBC AMVP mode.

[0350] In some embodiments, LIC parameters after being modified by a slope adjustment parameter are used with other LIC parameters. In some embodiments, the other LIC parameters are inherited LIC parameters. In some embodiments, a list of LIC parameters is constructed and a best LIC parameter is derived or signalled.

[0351] In some embodiments, reordering is used for the list of LIC parameters. In some embodiments, pruning is used for the list of LIC parameters.

[0352] In some embodiments, similar check is used for the list of LIC parameters, wherein a set LIC parameters is not added to the list of LIC parameters if the set of LIC parameters is similar to one of existing LIC parameters in the list of LIC parameters. In some embodiments, reordering is used for slope adjustment parameters.

[0353] In some embodiments, a template comprising neighbouring samples is used for the reordering, wherein different LIC parameters obtained using different slope adjustment parameters are used for the template, and a template cost is calculated and sort. In some embodiments, a template size is 1 or 2 or 3 or 4 or 5 or 6 or 7 or 8 or an adaptive integer.

[0354] In some embodiments, a slope adjustment parameter or an indication of the slope adjustment parameter is signalled for AMVP mode, and the slope adjustment parameter is used to modify the one or more parameters. In some embodiments, the target coding mode is not used for TM AMVP mode. In some embodiments, the target coding mode is not used to construct a TM based AMVP candidate list.

[0355] In some embodiments, the target coding mode is used for TM AMVP mode but not used to construct a TM based AMVP candidate list. In some embodiments, the merge mode comprises at least one of: regular merge mode, sub-block-based merge mode, TM merge mode, affine merge mode, MMVD, IBC merge mode, or GPM, or other merge modes.

[0356] In some embodiments, a slope adjustment parameter or an indication of the slope adjustment parameter is inherited from the merge mode, or whether the target coding mode is used is inherited from the merge mode, and the slope adjustment parameter is used to modify the one or more parameters. In some other embodiments, a slope adjustment parameter or a indication of the slope adjustment parameter is signaled for merge mode, or whether the target coding mode is used is signaled for merge mode. In some further embodiments, a slope adjustment parameter or an indication of the slope adjustment parameter is derived for merge mode, or whether the target coding mode is used is derived for merge mode.

[0357] In some embodiments, the target coding mode is applied to ARMC process.

Alternatively, the target coding mode is not applied to ARMC process. In some embodiments, the target coding mode is applied to ARMC process of a specific coding tool.

[0358] In some embodiments, the video unit comprises at least one of: a color component, a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding tree block (CTB), a coding unit (CU), a coding tree unit (CTU), a CTU row, groups of CTU, a slice, a tile, a sub-picture, a block, a sub-region within a block, or a region containing more than one sample or pixel.

[0359] In some embodiments, an indication of whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function is indicated at one of the followings: sequence level, group of pictures level, picture level, slice level, or tile group level.

[0360] In some embodiments, an indication of whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function 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.

[0361] In some embodiments, the method 4600 further comprises: determining whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function based on at least one of the followings: a message indicated in one of: DPS, SPS, VPS, PPS, APS, picture header, slice header, tile group header, largest coding unit (LCU), coding unit (CU), LCU row, group of LCUs, TU, PU block, video coding unit, a position of one of: CU, PU, TU, block, video coding unit, a block dimension of current block and/or its neighboring blocks, a block shape of current block and/or its neighboring blocks, a coded mode of the video unit, an indication of color format, a coding tree structure, a slice type, a tile group type, a picture type, a color component, a temporal layer identity, profiles or levels or Tiers of a standard.

[0362] In some embodiments, a syntax element is binarized as one of: a flag, a fixed length code, an EG(x) code, a unary code, a truncated unary code, or a truncated binary code. In some embodiments, a syntax element is coded with at least one context model, or wherein the syntax element bypass coded.

[0363] In some embodiments, a syntax element is signaled based on a condition. In some embodiments, the syntax element is signaled, only if a corresponding function is applicable, or wherein the syntax element is signaled, if dimensions of the video unit satisfy a condition.

[0364] In some embodiments, a syntax element is signaled at one of: block level, sequence level, group of pictures level, picture level, slice level, or tile group level. In some embodiments, the syntax element is signaled in one of the following: a coding structure of one of: CTU, CT, TU, PU, CTB, CB, TB, or PB, 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.

[0365] In some embodiments, the video unit is also applied with another coding tool. Alternatively, another coding tool is excluded for the video unit.

[0366] In some embodiments, the other coding tool comprises at least one of: affine, multiple transform selection (MTS), low-frequency non-separable transform (LFNST), merge mode with motion vector differences (MMVD), MIP, ISP, CCLM, CCCM, symmetric motion vector differences (SMVD), bi-directional optical flow (BDOF), DMVR, history-based motion vector predication (HMVP), template matching, IBC, or Palette.

[0367] In some embodiments, if the refined prediction sample of the prediction sample is derived by applying the function, the other coding tool is disabled implicitly without signaling. In some embodiments, if the other coding tool is used, deriving the refined prediction sample of the prediction sample by applying the function is disabled implicitly without signaling.

[0368] According to further embodiments of the present disclosure, a non-transitory computer-readable recording medium is provided. The non-transitory computer-readable recording medium stores a bitstream of a video which is generated by a method performed by an apparatus for video processing. The method comprises: deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction

mode or a coding tool; and generating the bitstream of the video unit based on the refined prediction sample.

[0369] According to still further embodiments of the present disclosure, a method for storing bitstream of a video is provided. The method comprises: deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; generating the bitstream of the video unit based on the refined prediction sample; and storing the bitstream in a non-transitory computer-readable recording medium.

[0370] 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.

[0371] Clause 1. A method for video processing, comprising: deriving, for a conversion between a video unit of a video and a bitstream of the video, a refined prediction sample of a prediction sample in the video unit by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and performing the conversion based on the refined prediction sample.

[0372] Clause 2. The method of clause 1, wherein the prediction mode comprises at least one of: an intra prediction mode, an inter prediction mode, an intra block copy (IBC) prediction mode, or other prediction mode.

[0373] Clause 3. The method of clause 1, wherein if the target coding mode is used, one or more coding tools are not used.

[0374] Clause 4. The method of clause 3, wherein overlapped block motion compensation (OBMC) is not used.

[0375] Clause 5. The method of clause 4, wherein an indication of OBMC is not signaled, if the target coding mode is used.

[0376] Clause 6. The method of clause 1, wherein the target coding mode is not allowed to be used for bi-prediction.

[0377] Clause 7. The method of clause 1, wherein at least one of: pre-defined adjustment parameter, signaled adjustment parameter or derived adjustment parameter is used for a block based reference picture derivation.

[0378] Clause 8. The method of clause 8, wherein the pre-defined adjustment parameter is used for the block based reference picture derivation.

[0379] Clause 9. The method of clause 8, wherein the pre-defined adjustment parameter is equal to one of: 1, -1, 2, -2, 3, or -3.

[0380] Clause 10. The method of clause 1, wherein the target coding mode is not used during a block based reference picture derivation.

[0381] Clause 11. The method of clause 1, wherein the target coding mode is applied to one or more inter coding tools, or wherein the target coding mode is not applied to one or more inter coding tools.

[0382] Clause 12. The method of clause 11, wherein the one or more inter coding tools comprise at least one of: a combined inter and intra prediction (CIIP), a variant of CIIP, a bi-prediction with coding unit-level weight (BCW), a variant of BCW a merge mode with motion vector difference (MMVD), a variant of MMVD, a template matching (TM), a variant of TM, an affine, a variant of affine, a decoder side motion vector refinement (DMVR), a variant of DMVR, a multi-pass DMVR, a prediction refinement with optical flow (PROF), a variant of PROF, a bi-directional optical flow (BDOF), a variant of BDOF, a sample based BDOF, an adaptive decoder-side motion vector refinement (ADMVR), a variant of ADMVR, an OBMC, a variant of OBMC, a TM based OBMC, a multi-hypothesis prediction (MHP), a variant of MHP, a geometric partitioning mode (GPM), a variant of GPM, a bilateral matching advanced motion vector prediction (AMVP)-merge mode, a template matching AMVP-merge mode, an Inter convolutional cross-component model (CCCM), an Inter cross-component prediction (CCP) merge mode, a reference picture resampling (RPR), or a variant of RPR.

[0383] Clause 13. The method of clause 12, wherein the CIIP comprises at least one of: CIIP-Planar, CIIP- template-based intra mode derivation (TIMD), or CIIP-TM, or wherein the BCW comprises BCW index derived by TM, or wherein the MMVD comprises a TM based reordering for MMVD, or wherein the affine comprises at least one of: affine-MMVD, TM based reordering for affine MMVD, or wherein the GPM comprises at least

one of: GPM-TM, GPM-MMVD, GPM-intra, GPM-Affine, GPM-IBC, regression-based GPM blending.

[0384] Clause 14. The method of clause 12, wherein the target coding mode is not used for RPR.

[0385] Clause 15. The method of clause 12, wherein if at least one of: BCW, adaptive motion vector resolution (AMVR), or symmetric motion vector differences (SMVD) is used, a syntax element indicating whether the target coding mode is used is not signaled.

[0386] Clause 16. The method of clause 12, wherein the target coding mode is not used for a cross-component coding tool which uses cross-component information to get the prediction or reconstruction.

[0387] Clause 17. The method of clause 16, wherein the cross-component coding tool comprises at least one of: a cross-component linear model (CCLM), a variant of CCLM, a multi-model CCLM (MMLM), a variant of MMLM, a gradient linear model (GLM), a variant of GLM, an inter convolutional cross-component model (CCCM), a variant of inter CCCM, a CCP merge, or a variant of CCP merge.

[0388] Clause 18. The method of clause 16, wherein the target coding mode is not used with the cross-component coding tool.

[0389] Clause 19. The method of clause 18, wherein if the target coding mode is used for the cross-component coding tool, parameters of LIC for luma and/or chroma components are not adjusted.

[0390] Clause 20. The method of clause 18, wherein if the target coding mode is used for the cross-component coding tool, LIC is disabled for luma and/or chroma components.

[0391] Clause 21. The method of clause 16, wherein if the target coding mode is used, a syntax element indicating whether the cross-component coding tool is used is not signalled.

[0392] Clause 22. The method of clause 16, wherein a determination of whether to use the target coding mode with the cross-component coding tool depends on information.

[0393] Clause 23. The method of clause 22, wherein the information comprises block dimensions or block size, and/or wherein the information comprises picture type or slice type.

[0394] Clause 24. The method of clause 23, wherein if the picture type if a low-delay picture or the slice type is a low-delay slice, the target coding mode is disallowed to be used with the cross-component coding tool.

[0395] Clause 25. The method of clause 11, wherein the target coding mode is not used for Affine mode.

[0396] Clause 26. The method of clause 11, wherein the target coding mode is not used for AMVP-merge mode.

[0397] Clause 27. The method of clause 11, wherein the target coding mode is not used for BCW.

[0398] Clause 28. The method of clause 11, wherein the target coding mode is not used for AMVR, or wherein the target coding mode is not used for a specific motion vector resolution.

[0399] Clause 29. The method of clause 11, wherein the target coding mode is not used for SMVD.

[0400] Clause 30. The method of clause 11, wherein the target coding mode is not used in a reordering process of motion vectors or motion candidates.

[0401] Clause 31. The method of clause 11, wherein the target coding mode is not used in the template matching, or a method which uses template matching to obtain the prediction.

[0402] Clause 32. The method of clause 11, wherein the coding tool comprises block-based delta pulse code modulation (BDPCM) or Palette.

[0403] Clause 33. The method of clause 11, wherein the coding tool comprises an IBC coding tool.

[0404] Clause 34. The method of clause 33, wherein the IBC coding tool comprises at least one of: an IBC AMVP mode, a variant of IBC AMVP, an IBC merge mode, a variant of IBC merge, an IBC-CIIP mode, a variant of IBC-CIIP, an IBC-GPM mode, a variant of IBC-GPM, an IBC-LIC mode, a variant of IBC-LIC, a sub-pel based IBC mode, a variant of sub-pel based IBC, a filtering based IBC mode, or a variant of filtering based IBC.

[0405] Clause 35. The method of clause 34, wherein the IBC AMVP mode comprises

at least one of a normal IBC AMVP mode, a TM based IBC AMVP mode, or a reconstruction-reordered IBC (RR-IBC), other IBC AMVP mode where a block vector (BV) predictor is derived and block vector difference (BVD) is signalled or derived, or wherein the IBC merge mode comprises at least one of a normal IBC merge mode, an IBC-TM merge mode, or an IBC- merge mode with block vector differences (IBC-MBVD) mode.

[0406] Clause 36. The method of clause 11, wherein the target coding mode is applied to uni-prediction and/or bi-prediction.

[0407] Clause 37. The method of clause 36, wherein if the target coding mode is applied to bi-prediction, one slope adjustment parameter is used.

[0408] Clause 38. The method of clause 11, wherein if the target coding mode is applied to bi-prediction, two slope adjustment parameters are used, wherein each slope adjustment is used to modify LIC parameters for one direction.

[0409] Clause 39. The method of clause 1, wherein the target coding mode is applied to AMVP mode, and/or wherein the target coding mode is applied to merge mode.

[0410] Clause 40. The method of clause 39, wherein the AMVP mode comprises at least one of: a regular AMVP mode, an affine AMVP mode, a TM AMVP mode, or an IBC AMVP mode.

[0411] Clause 41. The method of clause 39, wherein LIC parameters after being modified by a slope adjustment parameter are used with other LIC parameters.

[0412] Clause 42. The method of clause 41, wherein the other LIC parameters are inherited LIC parameters.

[0413] Clause 43. The method of clause 42, wherein a list of LIC parameters is constructed and a best LIC parameter is derived or signalled.

[0414] Clause 44. The method of clause 43, wherein reordering is used for the list of LIC parameters.

[0415] Clause 45. The method of clause 43, wherein pruning is used for the list of LIC parameters.

[0416] Clause 46. The method of clause 43, wherein similar check is used for the list of LIC parameters, wherein a set LIC parameters is not added to the list of LIC parameters

if the set of LIC parameters is similar to one of existing LIC parameters in the list of LIC parameters.

[0417] Clause 47. The method of clause 39, wherein reordering is used for slope adjustment parameters.

[0418] Clause 48. The method of clause 47, wherein a template comprising neighbouring samples is used for the reordering, wherein different LIC parameters obtained using different slope adjustment parameters are used for the template, and a template cost is calculated and sort.

[0419] Clause 49. The method of clause 48, wherein a template size is 1 or 2 or 3 or 4 or 5 or 6 or 7 or 8 or an adaptive integer.

[0420] Clause 50. The method of clause 39, wherein a slope adjustment parameter or an indication of the slope adjustment parameter is signalled for AMVP mode, and the slope adjustment parameter is used to modify the one or more parameters.

[0421] Clause 51. The method of clause 39, wherein the target coding mode is not used for TM AMVP mode.

[0422] Clause 52. The method of clause 51, wherein the target coding mode is not used to construct a TM based AMVP candidate list.

[0423] Clause 53. The method of clause 39, wherein the target coding mode is used for TM AMVP mode but not used to construct a TM based AMVP candidate list.

[0424] Clause 54. The method of clause 39, wherein the merge mode comprises at least one of: regular merge mode, sub-block-based merge mode, TM merge mode, affine merge mode, MMVD, IBC merge mode, or GPM, or other merge modes.

[0425] Clause 55. The method of clause 39, wherein a slope adjustment parameter or an indication of the slope adjustment parameter is inherited from the merge mode, or whether the target coding mode is used is inherited from the merge mode, and the slope adjustment parameter is used to modify the one or more parameters.

[0426] Clause 56. The method of clause 39, wherein a slope adjustment parameter or a indication of the slope adjustment parameter is signaled for merge mode, or whether the target coding mode is used is signaled for merge mode.

[0427] Clause 57. The method of clause 39, wherein a slope adjustment parameter or

an indication of the slope adjustment parameter is derived for merge mode, or whether the target coding mode is used is derived for merge mode.

[0428] Clause 58. The method of clause 39, wherein the target coding mode is applied to ARMC process, or wherein the target coding mode is not applied to ARMC process.

[0429] Clause 59. The method of clauses 58, wherein the target coding mode is applied to ARMC process of a specific coding tool.

[0430] Clause 60. The method of any of clauses 1-59, wherein the video unit comprises at least one of: a color component, a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding tree block (CTB), a coding unit (CU), a coding tree unit (CTU), a CTU row, groups of CTU, a slice, a tile, a sub-picture, a block, a sub-region within a block, or a region containing more than one sample or pixel.

[0431] Clause 61. The method of any of clauses 1-60, wherein an indication of whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function is indicated at one of the followings: sequence level, group of pictures level, picture level, slice level, or tile group level.

[0432] Clause 62. The method of any of clauses 1-60, wherein an indication of whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function 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.

[0433] Clause 63. The method of any of clauses 1-62, further comprising: determining whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function based on at least one of the followings: a message indicated in one of: DPS, SPS, VPS, PPS, APS, picture header, slice header, tile group header, largest coding unit (LCU), coding unit (CU), LCU row, group of LCUs, TU, PU block, video coding unit, a position of one of: CU, PU, TU, block, video coding unit, a block dimension of current block and/or its neighboring blocks, a block shape of current block and/or its neighboring blocks, a coded mode of the video unit, an indication of color

format, a coding tree structure, a slice type, a tile group type, a picture type, a color component, a temporal layer identity, profiles or levels or Tiers of a standard.

[0434] Clause 64.The method of any of clauses 1-63, wherein a syntax element is binarized as one of: a flag, a fixed length code, an EG(x) code, a unary code, a truncated unary code, or a truncated binary code.

[0435] Clause 65. The method of any of clauses 1-63, wherein a syntax element is coded with at least one context model, or wherein the syntax element bypass coded.

[0436] Clause 66. The method of any of clauses 1-63, wherein a syntax element is signaled based on a condition.

[0437] Clause 67. The method of clause 66, wherein the syntax element is signaled, only if a corresponding function is applicable, or wherein the syntax element is signaled, if dimensions of the video unit satisfy a condition.

[0438] Clause 68. The method of any of clauses 1-63, wherein a syntax element is signaled at one of: block level, sequence level, group of pictures level, picture level, slice level, or tile group level.

[0439] Clause 69. The method of clause 68, wherein the syntax element is signaled in one of the following: a coding structure of one of: CTU, CT, TU, PU, CTB, CB, TB, or PB, 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.

[0440] Clause 70. The method of any of clauses 1-69, wherein the video unit is also applied with another coding tool, or wherein another coding tool is excluded for the video unit.

[0441] Clause 71. The method of clause 70, wherein the other coding tool comprises at least one of: affine, multiple transform selection (MTS), low-frequency non-separable transform (LFNST), merge mode with motion vector differences (MMVD), MIP, ISP, CCLM, CCCM, symmetric motion vector differences (SMVD), bi-directional optical flow (BDOF), DMVR, history-based motion vector predication (HMVP), template matching, IBC, or Palette.

[0442] Clause 72. The method of clause 70, wherein if the refined prediction sample of the prediction sample is derived by applying the function, the other coding tool is disabled implicitly without signaling.

[0443] Clause 73. The method of clause 70, wherein if the other coding tool is used, deriving the refined prediction sample of the prediction sample by applying the function is disabled implicitly without signaling.

[0444] Clause 74. The method of any of clauses 1-73, wherein the conversion includes encoding the video unit into the bitstream.

[0445] Clause 75. The method of any of clauses 1-73, wherein the conversion includes decoding the video unit from the bitstream.

[0446] Clause 76. An apparatus for video processing 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 1-75.

[0447] Clause 77. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of clauses 1-75.

[0448] Clause 78. A non-transitory computer-readable recording medium storing a bitstream of a video which is generated by a method performed by an apparatus for video processing, wherein the method comprises: deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and generating the bitstream of the video unit based on the refined prediction sample.

[0449] Clause 79. A method for storing a bitstream of a video, comprising: deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least

one of: a prediction mode or a coding tool; generating the bitstream of the video unit based on the refined prediction sample; and storing the bitstream in a non-transitory computer-readable recording medium.

Example Device

[0450] Fig. 47 illustrates a block diagram of a computing device 4700 in which various embodiments of the present disclosure can be implemented. The computing device 4700 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).

[0451] It would be appreciated that the computing device 4700 shown in Fig. 47 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.

[0452] As shown in Fig. 47, the computing device 4700 includes a general-purpose computing device 4700. The computing device 4700 may at least comprise one or more processors or processing units 4710, a memory 4720, a storage unit 4730, one or more communication units 4740, one or more input devices 4750, and one or more output devices 4760.

[0453] In some embodiments, the computing device 4700 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 4700 can support any type of interface to a user (such as "wearable" circuitry and the like).

[0454] The processing unit 4710 may be a physical or virtual processor and can implement various processes based on programs stored in the memory 4720. 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 4700. The processing unit 4710 may also be referred to as a central processing unit (CPU), a microprocessor, a controller or a microcontroller.

[0455] The computing device 4700 typically includes various computer storage medium. Such medium can be any medium accessible by the computing device 4700, including, but not limited to, volatile and non-volatile medium, or detachable and non-detachable medium. The memory 4720 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 4730 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 4700.

[0456] The computing device 4700 may further include additional detachable/non-detachable, volatile/non-volatile memory medium. Although not shown in Fig. 47, 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.

[0457] The communication unit 4740 communicates with a further computing device via the communication medium. In addition, the functions of the components in the computing device 4700 can be implemented by a single computing cluster or multiple computing machines that can communicate via communication connections. Therefore, the computing device 4700 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.

[0458] The input device 4750 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 4760 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 4740, the computing device 4700 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 4700, or any devices (such as a network card, a modem and the like) enabling the computing device 4700 to communicate with one or more other computing devices, if required. Such communication can be performed via input/output (I/O) interfaces (not shown).

[0459] In some embodiments, instead of being integrated in a single device, some or all components of the computing device 4700 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.

[0460] The computing device 4700 may be used to implement video encoding/decoding in embodiments of the present disclosure. The memory 4720 may include one or more video coding modules 4725 having one or more program instructions. These modules are accessible and executable by the processing unit 4710 to perform the functionalities of the various embodiments described herein.

[0461] In the example embodiments of performing video encoding, the input device 4750 may receive video data as an input 4770 to be encoded. The video data may be processed, for example, by the video coding module 4725, to generate an encoded bitstream. The encoded bitstream may be provided via the output device 4760 as an output

4780.

[0462] In the example embodiments of performing video decoding, the input device 4750 may receive an encoded bitstream as the input 4770. The encoded bitstream may be processed, for example, by the video coding module 4725, to generate decoded video data. The decoded video data may be provided via the output device 4760 as the output 4780.

[0463] 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 for video processing, comprising:

deriving, for a conversion between a video unit of a video and a bitstream of the video, a refined prediction sample of a prediction sample in the video unit by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and

performing the conversion based on the refined prediction sample.

2. The method of claim 1, wherein the prediction mode comprises at least one of:

an intra prediction mode,

an inter prediction mode,

an intra block copy (IBC) prediction mode, or

other prediction mode.

3. The method of claim 1, wherein if the target coding mode is used, one or more coding tools are not used.

4. The method of claim 3, wherein overlapped block motion compensation (OBMC) is not used.

5. The method of claim 4, wherein an indication of OBMC is not signaled, if the target coding mode is used.

6. The method of claim 1, wherein the target coding mode is not allowed to be used for bi-prediction.

7. The method of claim 1, wherein at least one of: pre-defined adjustment parameter, signaled adjustment parameter or derived adjustment parameter is used for a block based reference picture derivation.

8. The method of claim 8, wherein the pre-defined adjustment parameter is used for the block based reference picture derivation.

9. The method of claim 8, wherein the pre-defined adjustment parameter is equal to one of: 1, -1, 2, -2, 3, or -3.

10. The method of claim 1, wherein the target coding mode is not used during a block based reference picture derivation.

11. The method of claim 1, wherein the target coding mode is applied to one or more inter coding tools, or

wherein the target coding mode is not applied to one or more inter coding tools.

12. The method of claim 11, wherein the one or more inter coding tools comprise at least one of: a combined inter and intra prediction (CIIP), a variant of CIIP, a bi-prediction with coding unit-level weight (BCW), a variant of BCW a merge mode with motion vector difference (MMVD), a variant of MMVD, a template matching (TM), a variant of TM, an affine, a variant of affine, a decoder side motion vector refinement (DMVR), a variant of DMVR, a multi-pass DMVR, a prediction refinement with optical flow (PROF), a variant of PROF, a bi-directional optical flow (BDOF), a variant of BDOF, a sample based BDOF, an adaptive decoder-side motion vector refinement (ADMVR), a variant of ADMVR, an OBMC, a variant of OBMC, a TM based OBMC, a multi-hypothesis prediction (MHP), a variant of MHP, a geometric partitioning mode (GPM), a variant of GPM, a bilateral matching advanced motion vector prediction (AMVP)-merge mode, a template matching AMVP-merge mode, an Inter convolutional cross-component model (CCCM), an Inter cross-component prediction (CCP) merge mode, a reference picture resampling (RPR), or a variant of RPR.

13. The method of claim 12, wherein the CIIP comprises at least one of: CIIP-Planar, CIIP- template-based intra mode derivation (TIMD), or CIIP-TM, or

wherein the BCW comprises BCW index derived by TM, or

wherein the MMVD comprises a TM based reordering for MMVD, or

wherein the affine comprises at least one of: affine-MMVD, TM based reordering for affine MMVD, or

wherein the GPM comprises at least one of: GPM-TM, GPM-MMVD, GPM-intra, GPM-Affine, GPM-IBC, regression-based GPM blending.

14. The method of claim 12, wherein the target coding mode is not used for RPR.

15. The method of claim 12, wherein if at least one of: BCW, adaptive motion vector resolution (AMVR), or symmetric motion vector differences (SMVD) is used, a syntax element indicating whether the target coding mode is used is not signaled.

16. The method of claim 12, wherein the target coding mode is not used for a cross-component coding tool which uses cross-component information to get the prediction or reconstruction.

17. The method of claim 16, wherein the cross-component coding tool comprises at least one of: a cross-component linear model (CCLM), a variant of CCLM, a multi-model CCLM (MMLM), a variant of MMLM, a gradient linear model (GLM), a variant of GLM, an inter convolutional cross-component model (CCCM), a variant of inter CCCM, a CCP merge, or a variant of CCP merge.

18. The method of claim 16, wherein the target coding mode is not used with the cross-component coding tool.

19. The method of claim 18, wherein if the target coding mode is used for the cross-component coding tool, parameters of LIC for luma and/or chroma components are not adjusted.

20. The method of claim 18, wherein if the target coding mode is used for the cross-component coding tool, LIC is disabled for luma and/or chroma components.

21. The method of claim 16, wherein if the target coding mode is used, a syntax element indicating whether the cross-component coding tool is used is not signalled.

22. The method of claim 16, wherein a determination of whether to use the target coding mode with the cross-component coding tool depends on information.

23. The method of claim 22, wherein the information comprises block dimensions or block size, and/or

wherein the information comprises picture type or slice type.

24. The method of claim 23, wherein if the picture type if a low-delay picture or the slice type is a low-delay slice, the target coding mode is disallowed to be used with the cross-component coding tool.

25. The method of claim 11, wherein the target coding mode is not used for Affine mode.

26. The method of claim 11, wherein the target coding mode is not used for AMVP-merge mode.

27. The method of claim 11, wherein the target coding mode is not used for BCW.

28. The method of claim 11, wherein the target coding mode is not used for AMVR, or wherein the target coding mode is not used for a specific motion vector resolution.

29. The method of claim 11, wherein the target coding mode is not used for SMVD.

30. The method of claim 11, wherein the target coding mode is not used in a reordering process of motion vectors or motion candidates.

31. The method of claim 11, wherein the target coding mode is not used in the template matching, or a method which uses template matching to obtain the prediction.

32. The method of claim 11, wherein the coding tool comprises block-based delta pulse code modulation (BDPCM) or Palette.

33. The method of claim 11, wherein the coding tool comprises an IBC coding tool.

34. The method of claim 33, wherein the IBC coding tool comprises at least one of:

an IBC AMVP mode,
a variant of IBC AMVP,
an IBC merge mode,
a variant of IBC merge,
an IBC-CIIP mode,
a variant of IBC-CIIP,
an IBC-GPM mode,
a variant of IBC-GPM,
an IBC-LIC mode,
a variant of IBC-LIC,
a sub-pel based IBC mode,
a variant of sub-pel based IBC,
a filtering based IBC mode, or
a variant of filtering based IBC.

35. The method of claim 34, wherein the IBC AMVP mode comprises at least one of a normal IBC AMVP mode, a TM based IBC AMVP mode, or a reconstruction-reordered IBC (RR-IBC), other IBC AMVP mode where a block vector (BV) predictor is derived and block vector difference (BVD) is signalled or derived, or

wherein the IBC merge mode comprises at least one of a normal IBC merge mode, an IBC-TM merge mode, or an IBC- merge mode with block vector differences (IBC-MBVD) mode.

36. The method of claim 11, wherein the target coding mode is applied to uni-prediction and/or bi-prediction.

37. The method of claim 36, wherein if the target coding mode is applied to bi-prediction, one slope adjustment parameter is used.

38. The method of claim 11, wherein if the target coding mode is applied to bi-prediction, two slope adjustment parameters are used, wherein each slope adjustment is used to modify LIC parameters for one direction.

39. The method of claim 1, wherein the target coding mode is applied to AMVP mode, and/or

wherein the target coding mode is applied to merge mode.

40. The method of claim 39, wherein the AMVP mode comprises at least one of: a regular AMVP mode, an affine AMVP mode, a TM AMVP mode, or an IBC AMVP mode.

41. The method of claim 39, wherein LIC parameters after being modified by a slope adjustment parameter are used with other LIC parameters.

42. The method of claim 41, wherein the other LIC parameters are inherited LIC parameters.

43. The method of claim 42, wherein a list of LIC parameters is constructed and a best LIC parameter is derived or signalled.

44. The method of claim 43, wherein reordering is used for the list of LIC parameters.

45. The method of claim 43, wherein pruning is used for the list of LIC parameters.

46. The method of claim 43, wherein similar check is used for the list of LIC parameters, wherein a set LIC parameters is not added to the list of LIC parameters if the set of LIC parameters is similar to one of existing LIC parameters in the list of LIC parameters.

47. The method of claim 39, wherein reordering is used for slope adjustment parameters.

48. The method of claim 47, wherein a template comprising neighbouring samples is used for the reordering, wherein different LIC parameters obtained using different slope adjustment parameters are used for the template, and a template cost is calculated and sort.

49. The method of claim 48, wherein a template size is 1 or 2 or 3 or 4 or 5 or 6 or 7 or 8 or an adaptive integer.

50. The method of claim 39, wherein a slope adjustment parameter or an indication of the slope adjustment parameter is signalled for AMVP mode, and the slope adjustment parameter is used to modify the one or more parameters.

51. The method of claim 39, wherein the target coding mode is not used for TM AMVP mode.

52. The method of claim 51, wherein the target coding mode is not used to construct a TM based AMVP candidate list.

53. The method of claim 39, wherein the target coding mode is used for TM AMVP mode but not used to construct a TM based AMVP candidate list.

54. The method of claim 39, wherein the merge mode comprises at least one of: regular merge mode, sub-block-based merge mode, TM merge mode, affine merge mode, MMVD, IBC merge mode, or GPM, or other merge modes.

55. The method of claim 39, wherein a slope adjustment parameter or an indication of the slope adjustment parameter is inherited from the merge mode, or whether the target coding mode is used is inherited from the merge mode, and the slope adjustment parameter is used to modify the one or more parameters.

56. The method of claim 39, wherein a slope adjustment parameter or an indication of the slope adjustment parameter is signaled for merge mode, or whether the target coding mode is used is signaled for merge mode.

57. The method of claim 39, wherein a slope adjustment parameter or an indication of the slope adjustment parameter is derived for merge mode, or whether the target coding mode is used is derived for merge mode.

58. The method of claim 39, wherein the target coding mode is applied to ARMC process, or

wherein the target coding mode is not applied to ARMC process.

59. The method of claims 58, wherein the target coding mode is applied to ARMC process of a specific coding tool.

60. The method of any of claims 1-59, wherein the video unit comprises at least one of:

a color component,
a prediction block (PB),
a transform block (TB),
a coding block (CB),
a prediction unit (PU),
a transform unit (TU),
a coding tree block (CTB),
a coding unit (CU),
a coding tree unit (CTU),
a CTU row,
groups of CTU,
a slice,
a tile,
a sub-picture,
a block,
a sub-region within a block, or
a region containing more than one sample or pixel.

61. The method of any of claims 1-60, wherein an indication of whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function is indicated at one of the followings:

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

62. The method of any of claims 1-60, wherein an indication of whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function 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.

63. The method of any of claims 1-62, further comprising:

determining whether to and/or how to derive the refined prediction sample of the prediction sample in the video unit by applying the function based on at least one of the followings:

a message indicated in one of: DPS, SPS, VPS, PPS, APS, picture header, slice header, tile group header, largest coding unit (LCU), coding unit (CU), LCU row, group of LCUs, TU, PU block, video coding unit,

a position of one of: CU, PU, TU, block, video coding unit,

a block dimension of current block and/or its neighboring blocks,

a block shape of current block and/or its neighboring blocks,

a coded mode of the video unit,

an indication of color format,

a coding tree structure,

a slice type,

a tile group type,

a picture type,

a color component,

a temporal layer identity,

profiles or levels or Tiers of a standard.

64.The method of any of claims 1-63, wherein a syntax element is binarized as one of: a flag, a fixed length code, an EG(x) code, a unary code, a truncated unary code, or a truncated binary code.

65. The method of any of claims 1-63, wherein a syntax element is coded with at least one context model, or

wherein the syntax element bypass coded.

66. The method of any of claims 1-63, wherein a syntax element is signaled based on a condition.

67. The method of claim 66, wherein the syntax element is signaled, only if a corresponding function is applicable, or

wherein the syntax element is signaled, if dimensions of the video unit satisfy a condition.

68. The method of any of claims 1-63, wherein a syntax element is signaled at one of:

block level,

sequence level,

group of pictures level,

picture level,

slice level, or

tile group level.

69. The method of claim 68, wherein the syntax element is signaled in one of the following:

a coding structure of one of: CTU, CT, TU, PU, CTB, CB, TB, or PB,

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.

70. The method of any of claims 1-69, wherein the video unit is also applied with another coding tool, or

wherein another coding tool is excluded for the video unit.

71. The method of claim 70, wherein the other coding tool comprises at least one of: affine, multiple transform selection (MTS), low-frequency non-separable transform (LFNST), merge mode with motion vector differences (MMVD), MIP, ISP, CCLM, CCCM, symmetric motion vector differences (SMVD), bi-directional optical flow (BDOF), DMVR, history-based motion vector predication (HMVP), template matching, IBC, or Palette.

72. The method of claim 70, wherein if the refined prediction sample of the prediction sample is derived by applying the function, the other coding tool is disabled implicitly without signaling.

73. The method of claim 70, wherein if the other coding tool is used, deriving the refined prediction sample of the prediction sample by applying the function is disabled implicitly without signaling.

74. The method of any of claims 1-73, wherein the conversion includes encoding the video unit into the bitstream.

75. The method of any of claims 1-73, wherein the conversion includes decoding the video unit from the bitstream.

76. An apparatus for video processing 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 1-75.

77. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of claims 1-75.

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

deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool; and

generating the bitstream of the video unit based on the refined prediction sample.

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

deriving a refined prediction sample of a prediction sample in a video unit of the video by applying a function used in local illumination compensation (LIC) to the prediction sample, in response to that the video unit is coded with a target coding mode, wherein one or more parameters of the function are modified and the target coding mode is applied to at least one of: a prediction mode or a coding tool;

generating the bitstream of the video unit based on the refined prediction sample; and

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

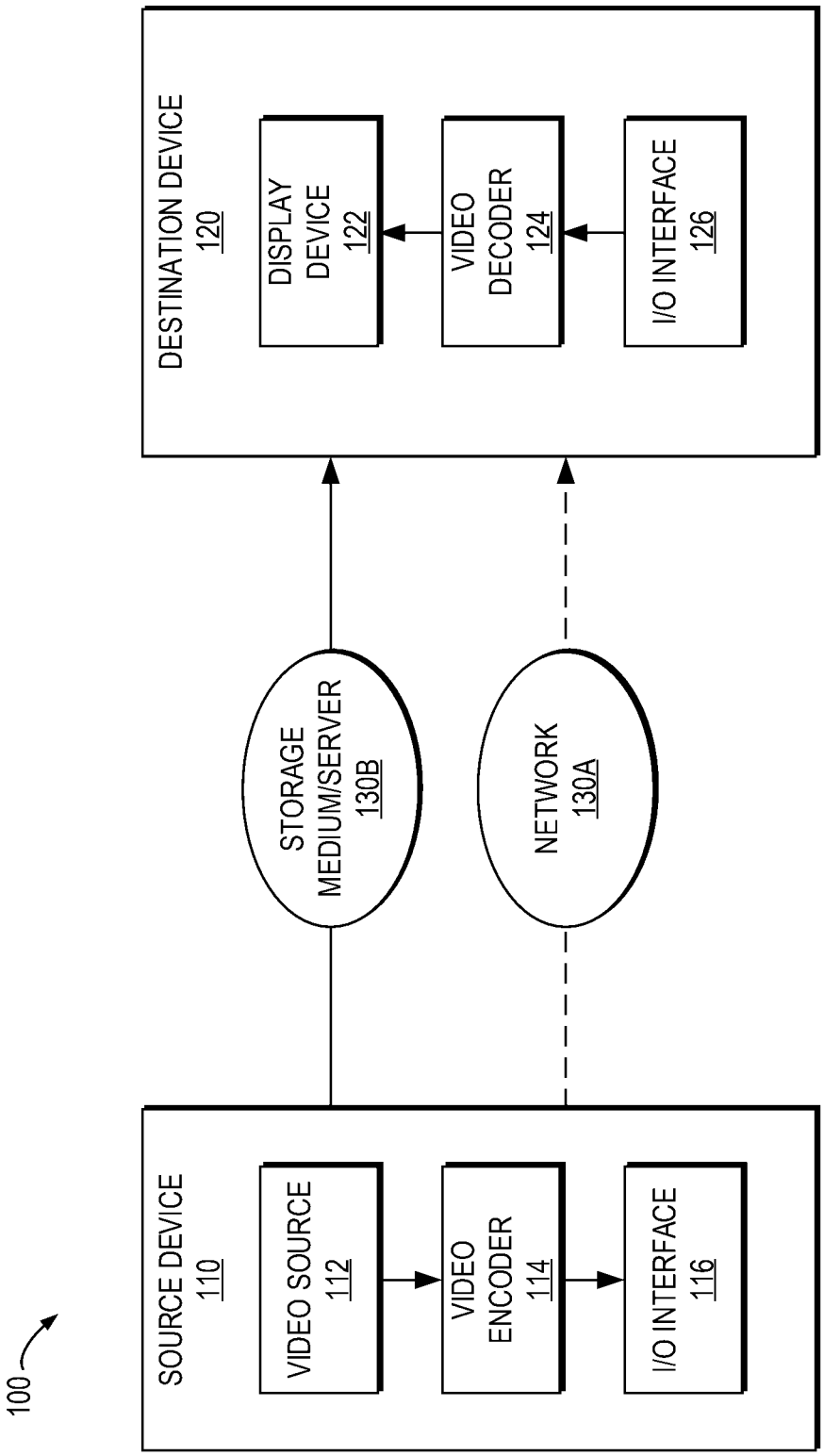
100
SOURCE DEVICE
110
VIDEO SOURCE
112
VIDEO
ENCODER
114
I/O INTERFACE
116
STORAGE
MEDIUM/SERVER
130B
NETWORK
130A
DESTINATION DEVICE
120
DISPLAY
DEVICE
122
VIDEO
DECODER
124
I/O INTERFACE
126

Fig. 1

5

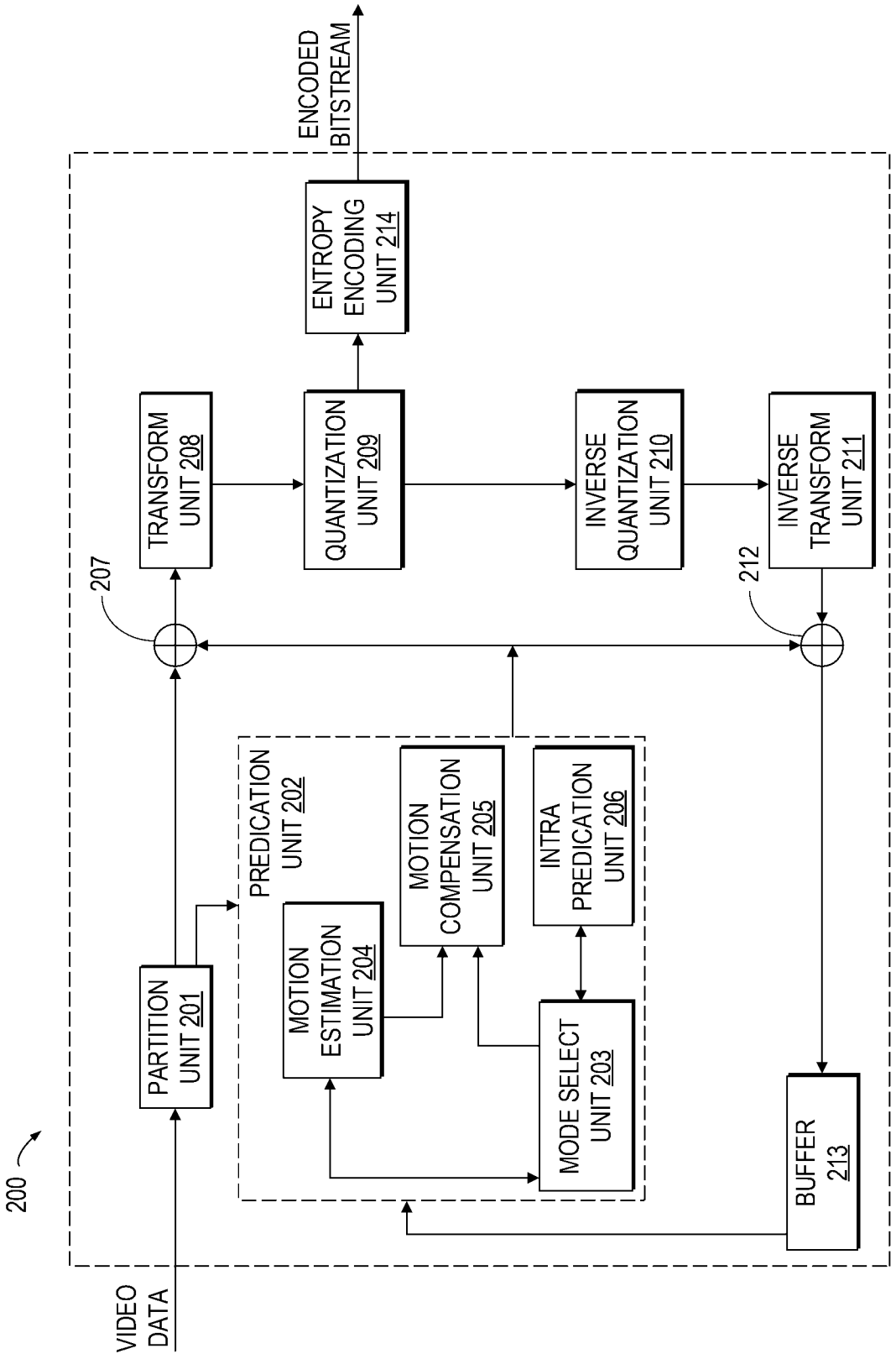
200
VIDEO DATA
PARTITION UNIT 201
PREDICATION UNIT 202
MOTION ESTIMATION UNIT 204
MOTION COMPENSATION UNIT 205
INTRA PREDICATION UNIT 206
MODE SELECT UNIT 203
207
TRANSFORM UNIT 208
QUANTIZATION UNIT 209
ENTROPY ENCODING UNIT 214
ENCODED BITSTREAM
INVERSE QUANTIZATION UNIT 210
INVERSE TRANSFORM UNIT 211
212
BUFFER 213

Fig. 2

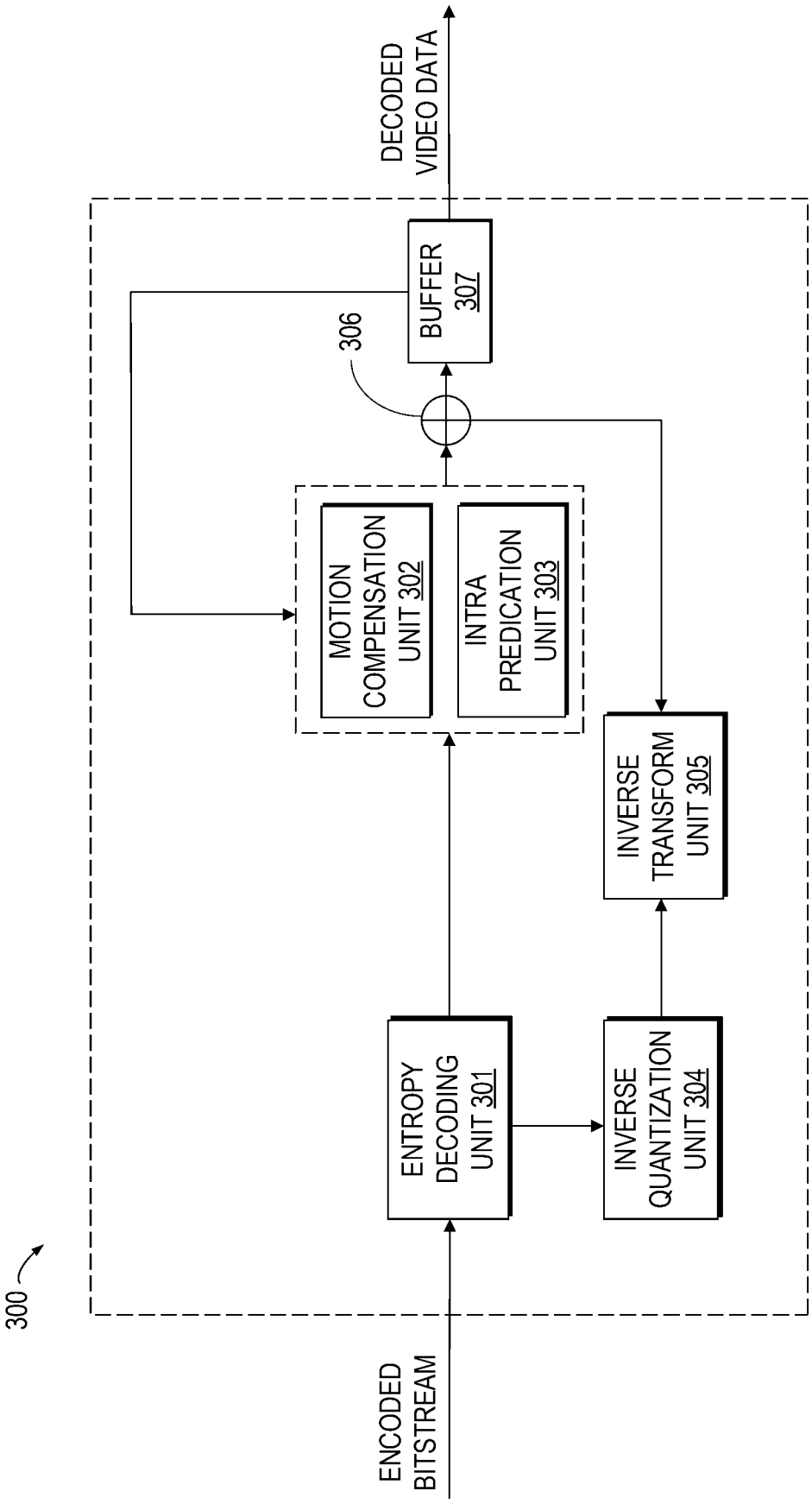
300
ENCODED BITSTREAM
ENTROPY DECODING UNIT 301
INVERSE QUANTIZATION UNIT 304
INVERSE TRANSFORM UNIT 305
MOTION COMPENSATION UNIT 302
INTRA PREDICATION UNIT 303
306
BUFFER 307
DECODED VIDEO DATA

Fig. 3

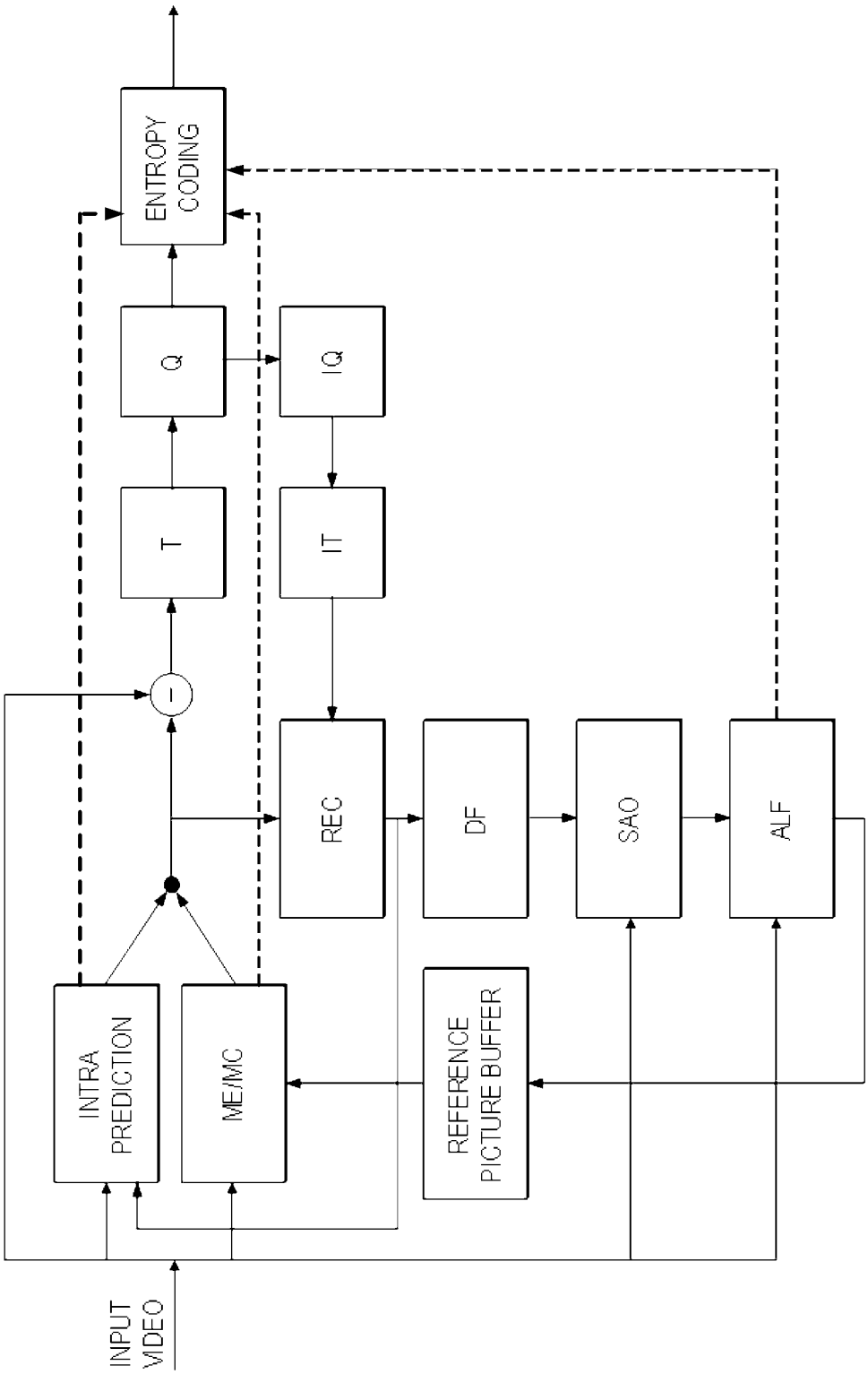
ENTROPY CODING
Q
IQ
T
IT
REC
DF
SAO
ALF
INTRA PREDICTION
ME/MC
REFERENCE PICTURE BUFFER
INPUT VIDEO

Fig. 4

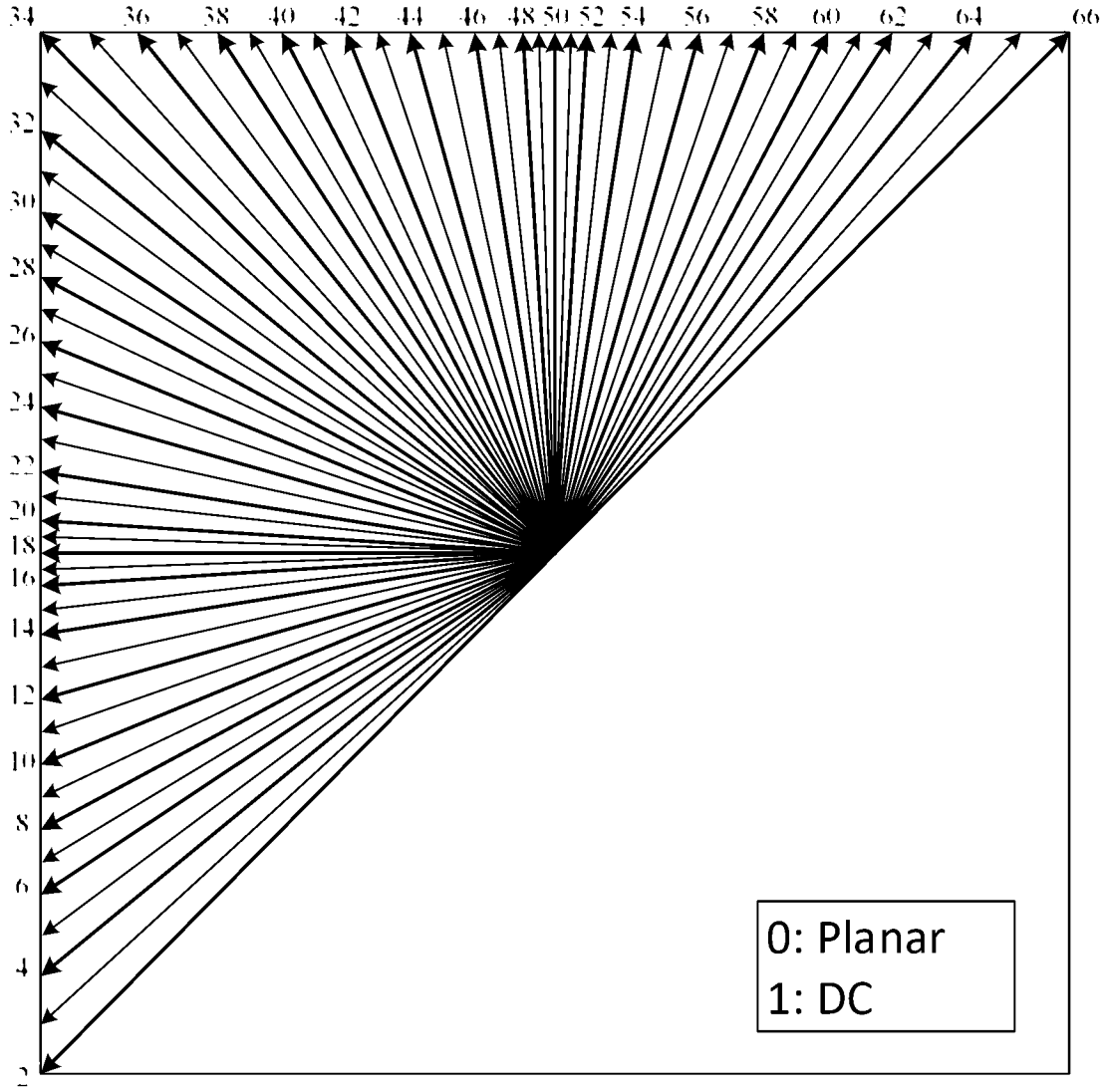
34
36
38
40
42
44
46
48
50
52
54
56
58
60
62
64
66
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0: Planar
1: DC

Fig. 5

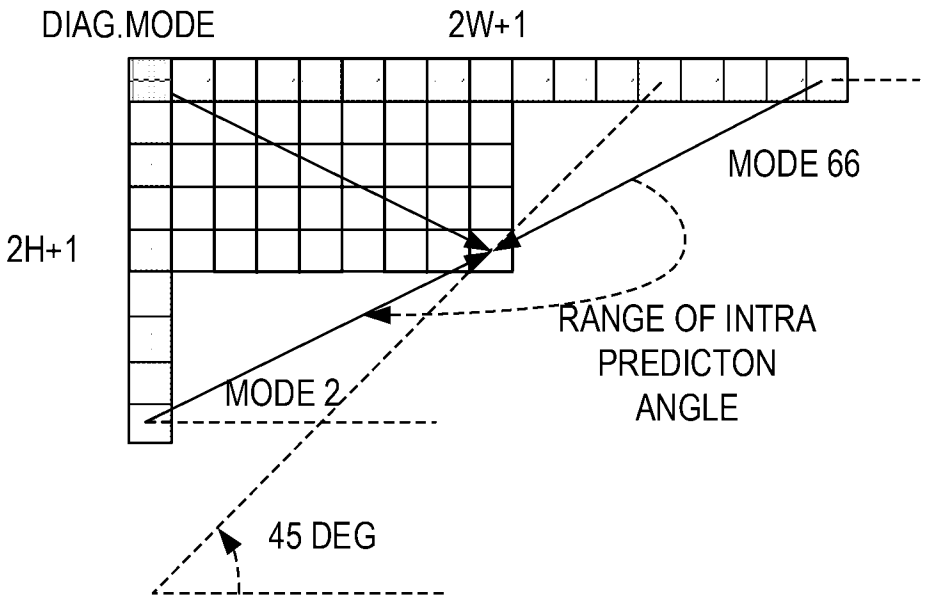
DIAG.MODE
2W+1
MODE 66
2H+1
RANGE OF INTRA
PREDICTON
ANGLE
MODE 2
45 DEG

Fig. 6A

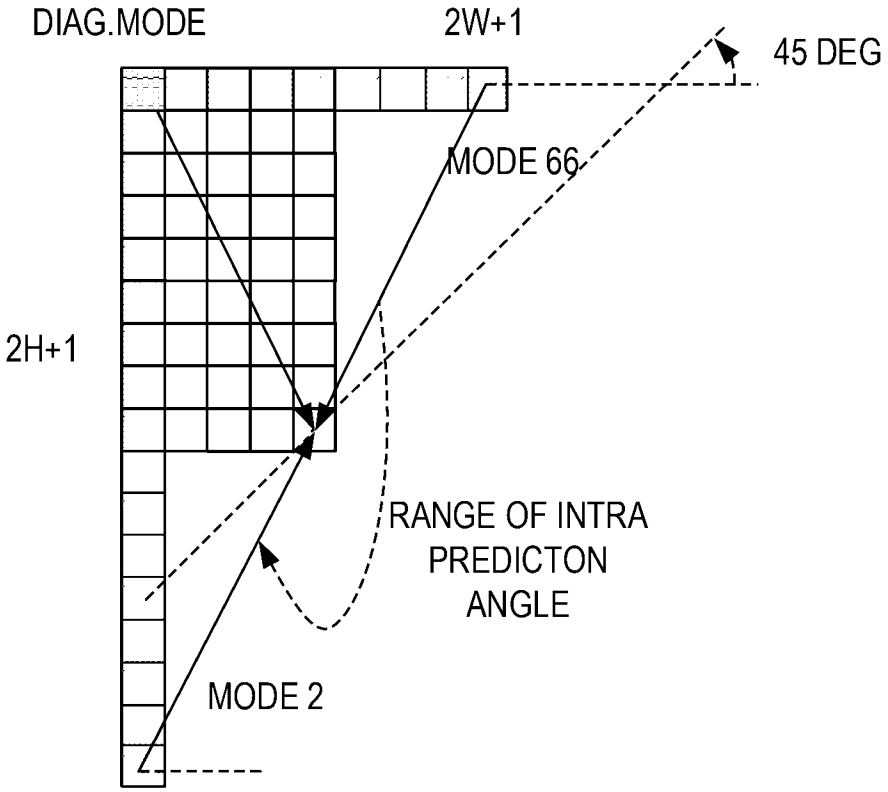
DIAG.MODE
2W+1
45 DEG
MODE 66
2H+1
RANGE OF INTRA
PREDICTON
ANGLE
MODE 2

Fig. 6B

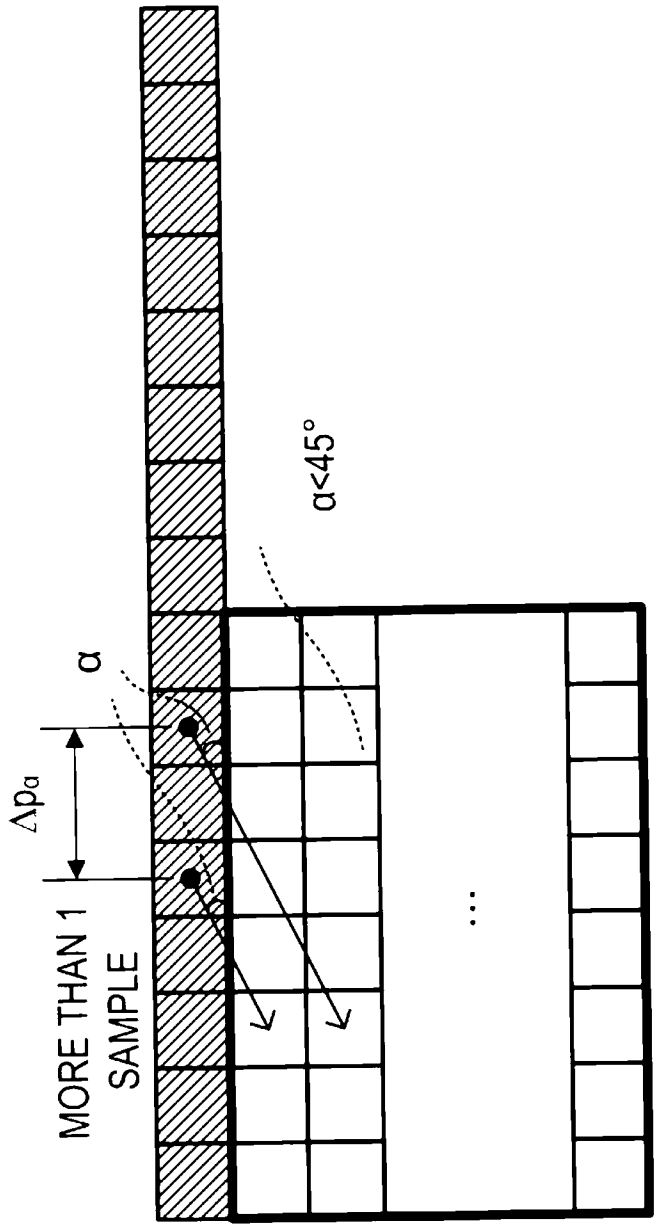

Fig. 7

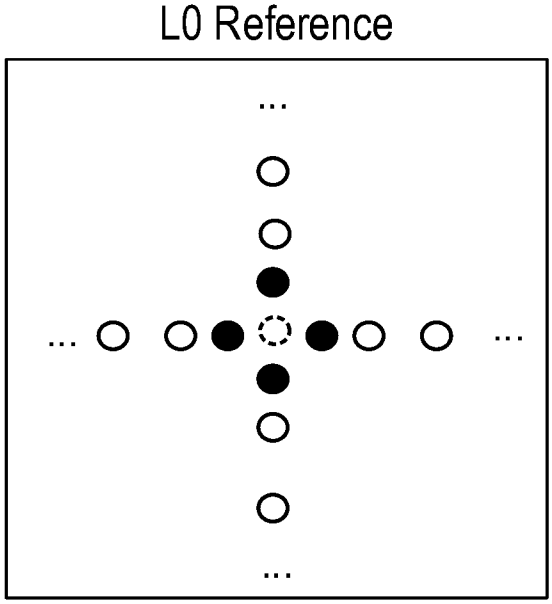

Fig. 8A

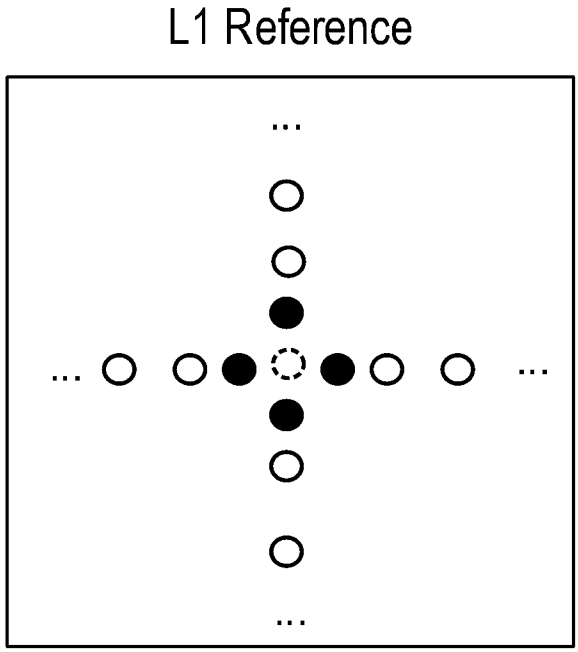

Fig. 8B

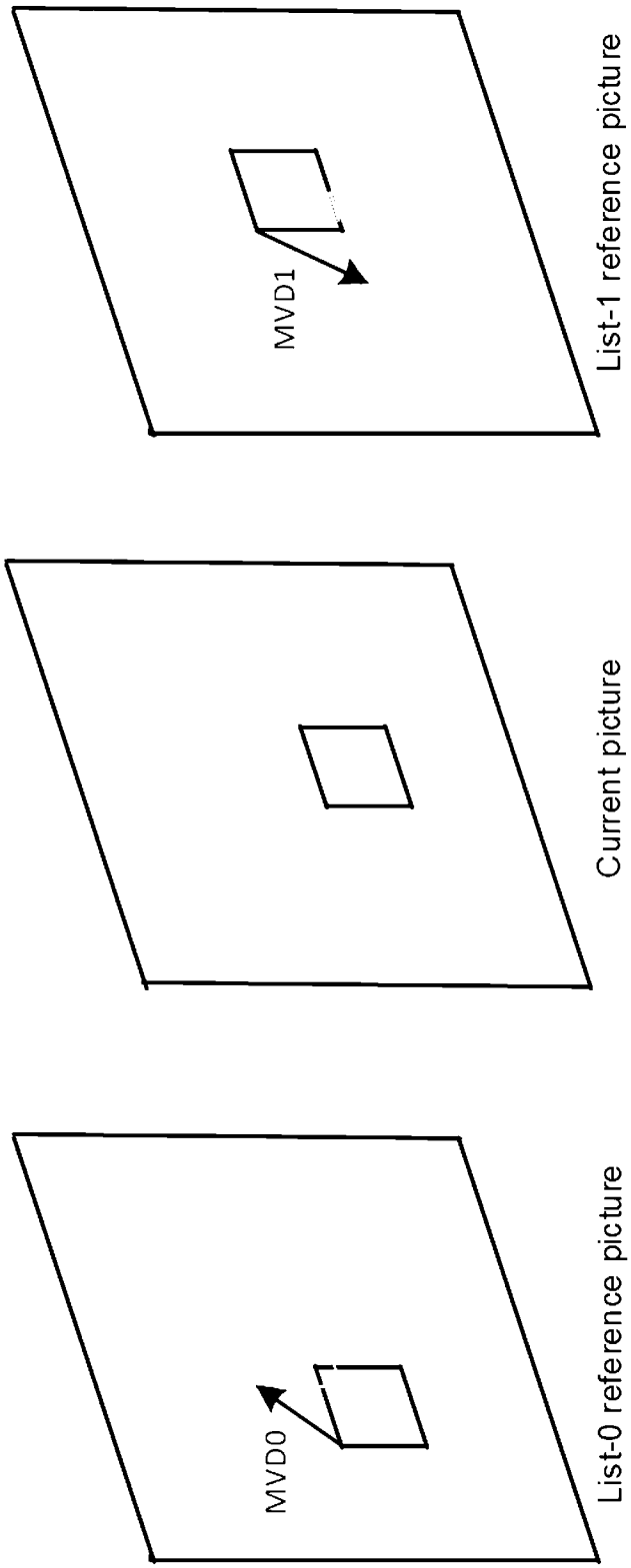
List-1 reference picture
MVD1
Current picture
List-0 reference picture
MVD0

Fig. 9

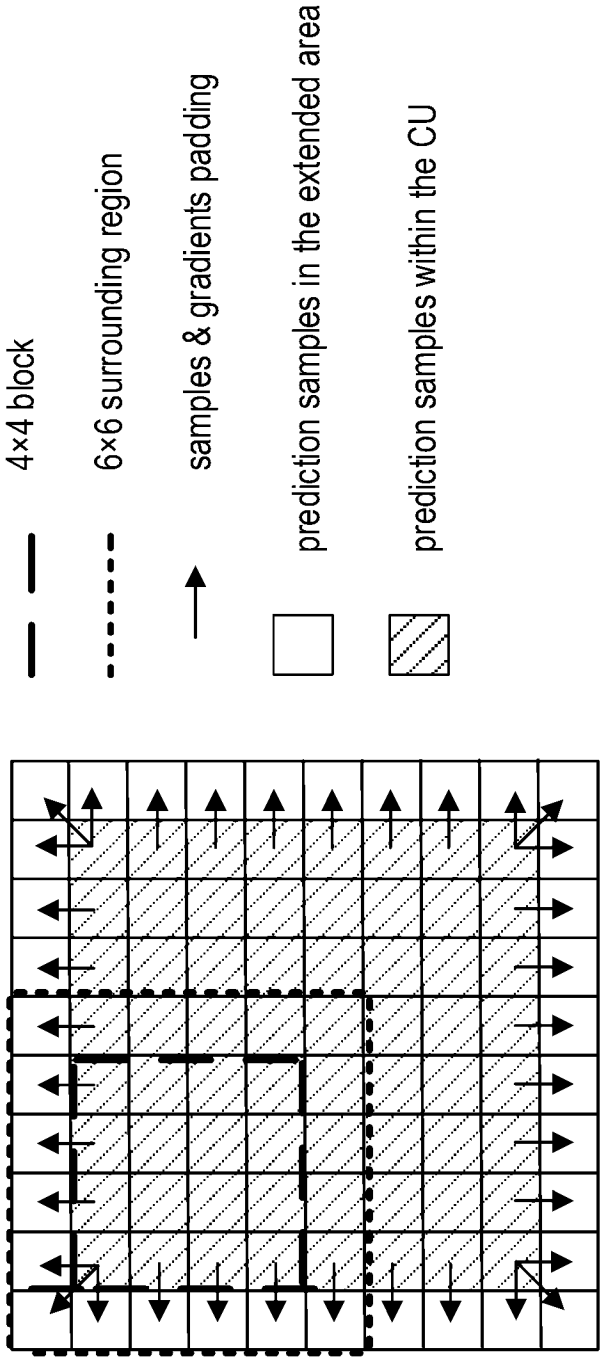
4×4 block
6×6 surrounding region
samples & gradients padding
prediction samples in the extended area
prediction samples within the CU

Fig. 10

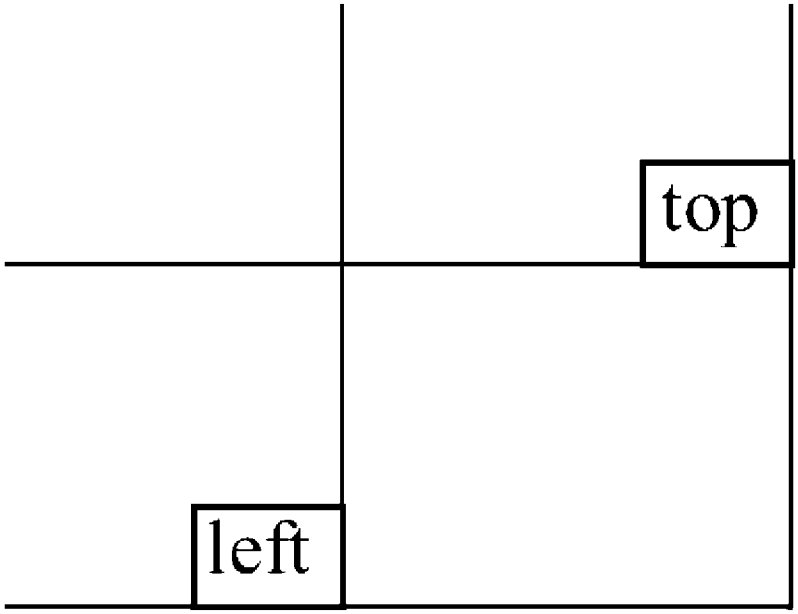

Fig. 11

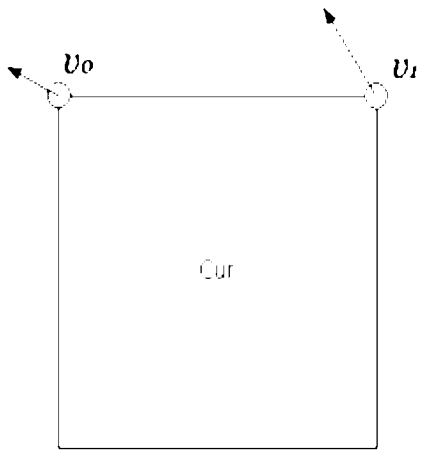

Fig. 12A

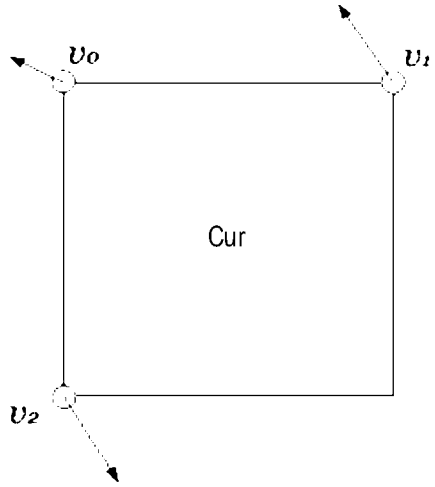

Fig. 12B

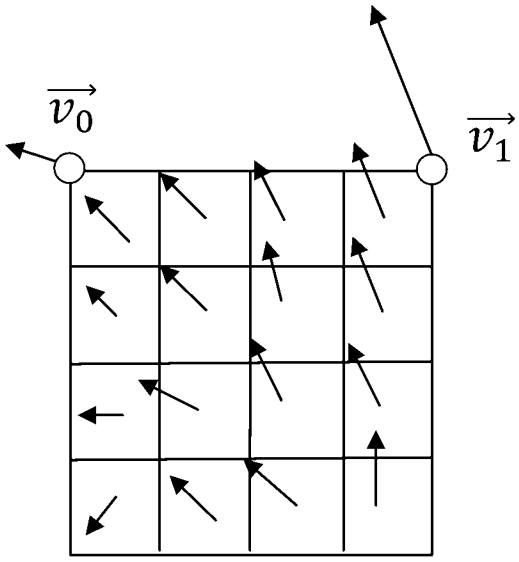

Fig. 13

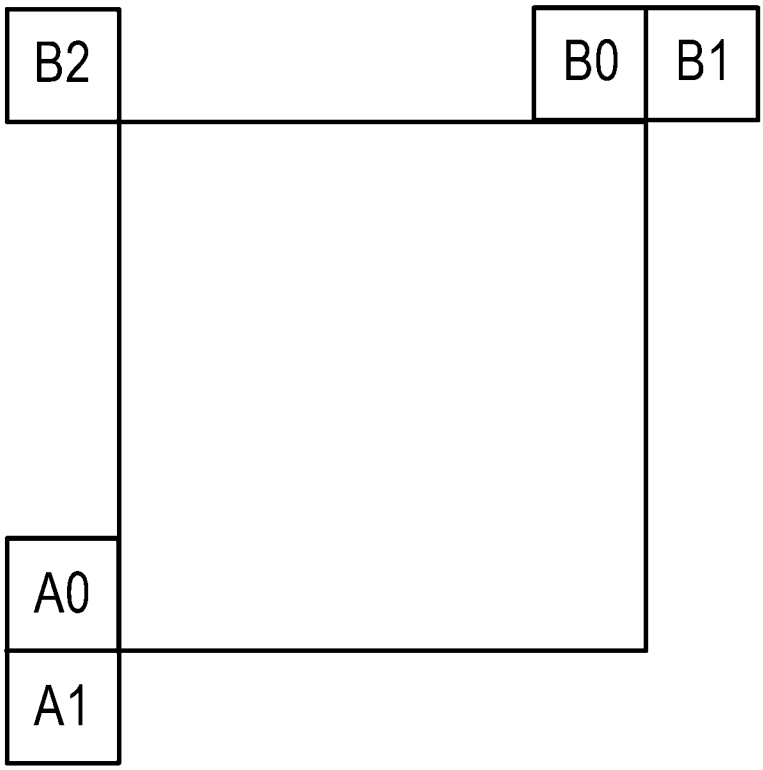

Fig. 14

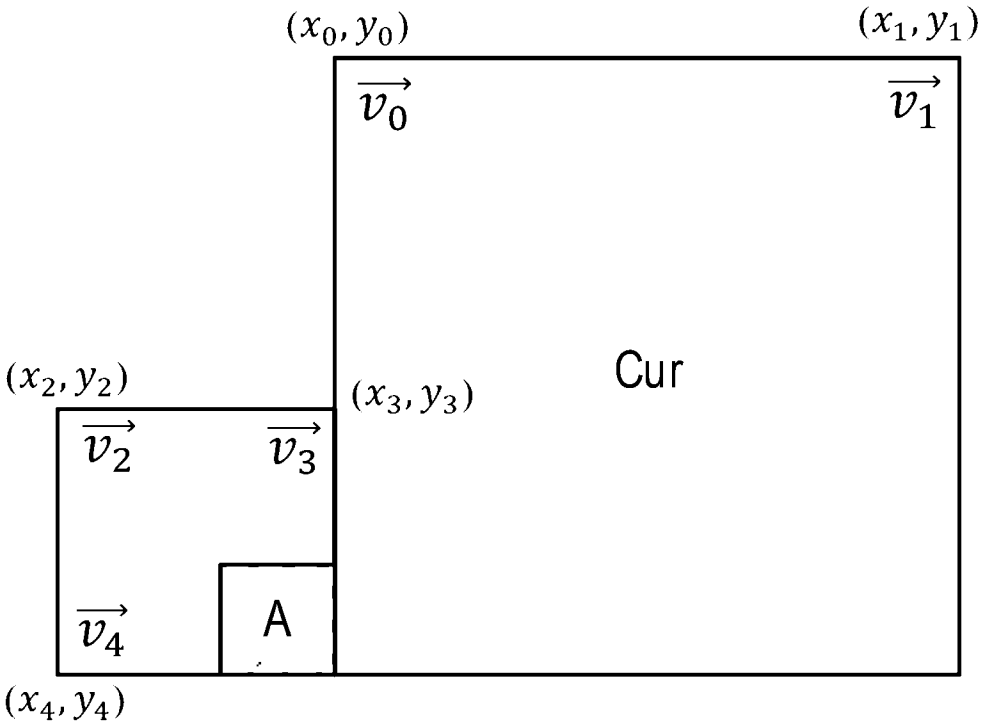
(x_0, y_0)
(x_1, y_1)
$\overrightarrow{v_0}$
$\overrightarrow{v_1}$
Cur
(x_2, y_2)
(x_3, y_3)
$\overrightarrow{v_2}$
$\overrightarrow{v_3}$
A
$\overrightarrow{v_4}$
(x_4, y_4)

Fig. 15

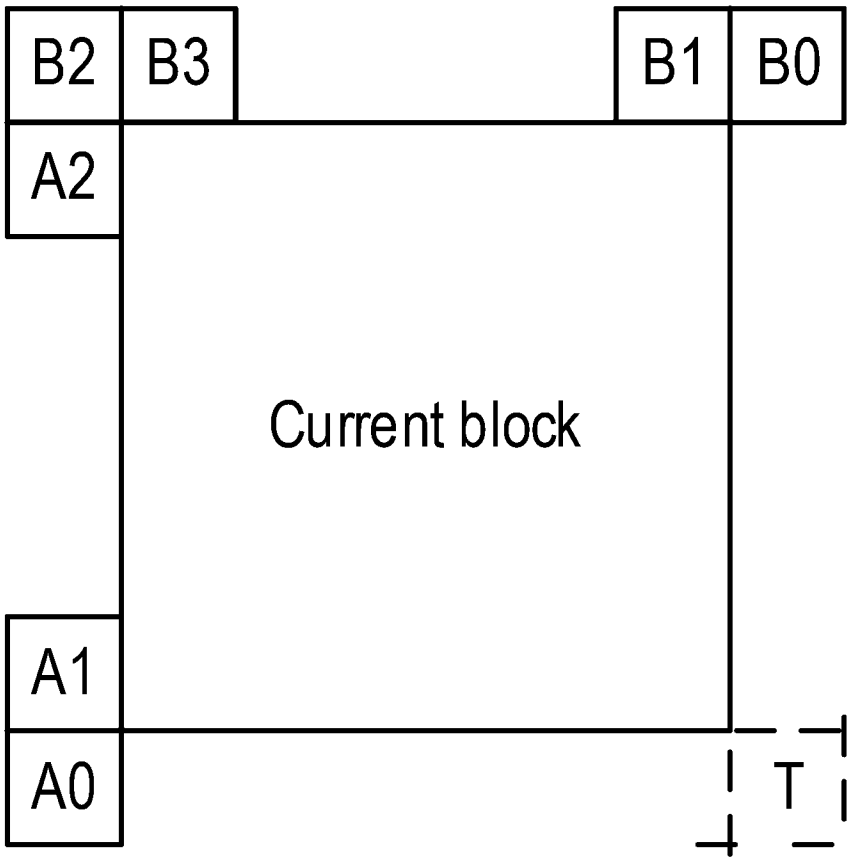

Fig. 16

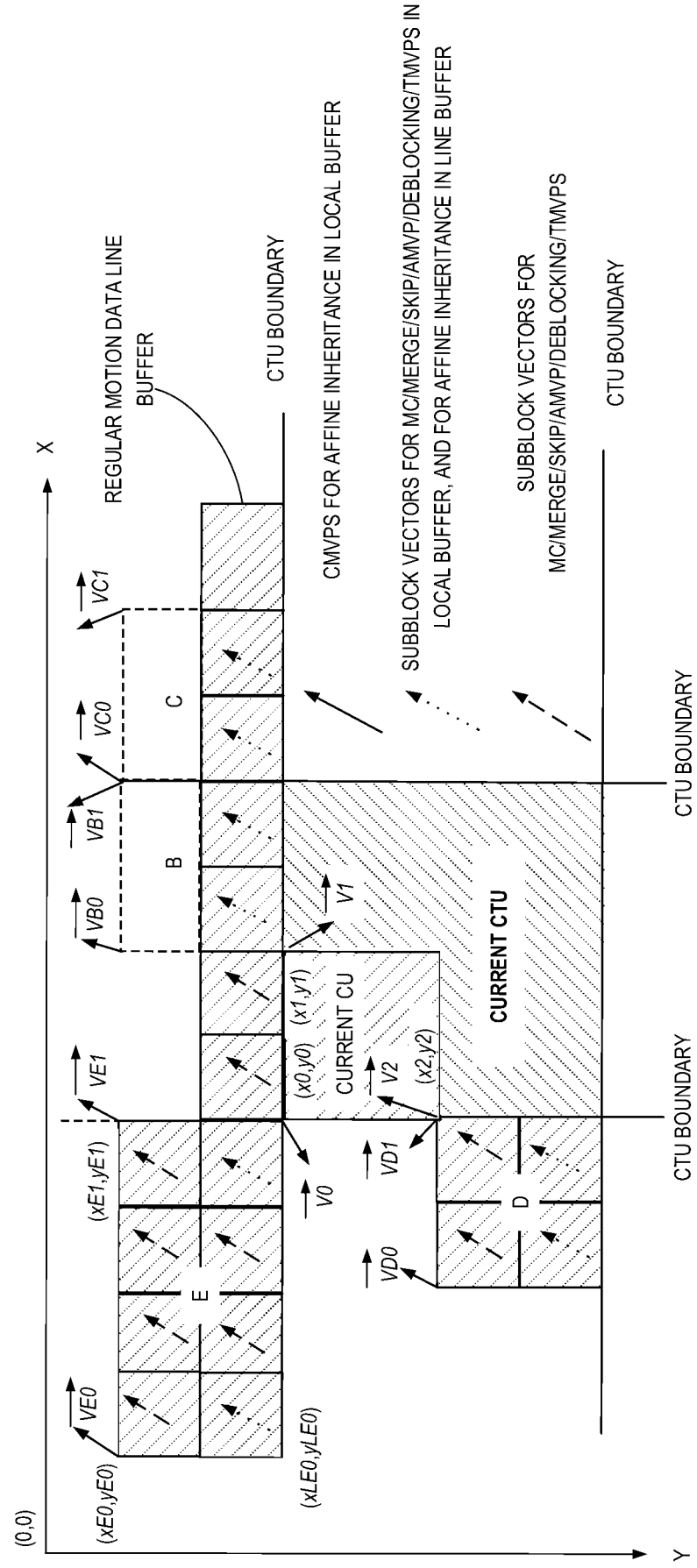
X
Y
(0,0)
REGULAR MOTION DATA LINE BUFFER
CTU BOUNDARY
CMVPS FOR AFFINE INHERITANCE IN LOCAL BUFFER
SUBBLOCK VECTORS FOR MC/MERGE/SKIP/AMVP/DEBLOCKING/TMVPS IN LOCAL BUFFER, AND FOR AFFINE INHERITANCE IN LINE BUFFER
SUBBLOCK VECTORS FOR MC/MERGE/SKIP/AMVP/DEBLOCKING/TMVPS
CTU BOUNDARY
CTU BOUNDARY
CTU BOUNDARY
CURRENT CTU
CURRENT CU
A
B
C
D
E
VE0
VE1
VB0
VB1
VC0
VC1
VD0
VD1
V0
V1
V2
(xE0,yE0)
(xE1,yE1)
(xLE0,yLE0)
(x0,y0)
(x1,y1)
(x2,y2)

Fig. 17

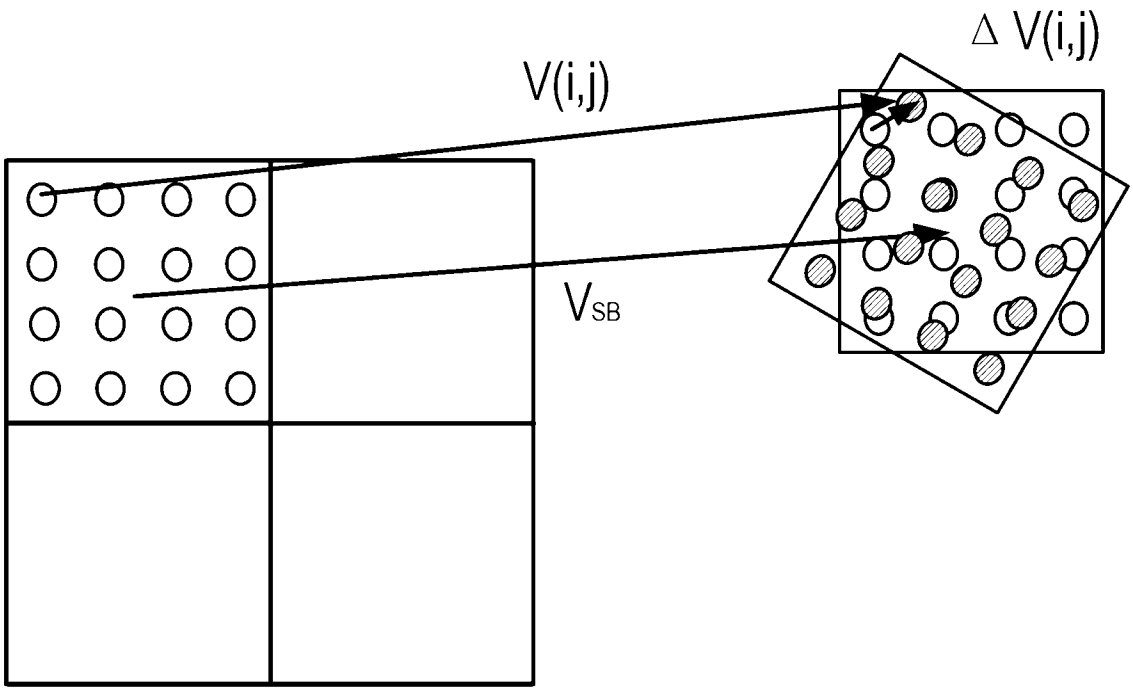

Fig. 18

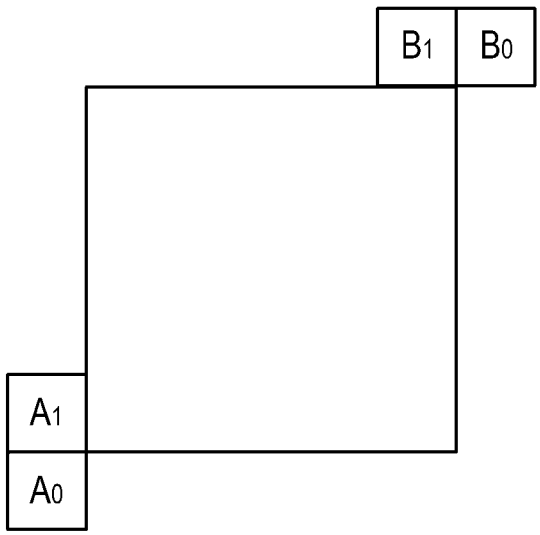

Fig. 19A

5

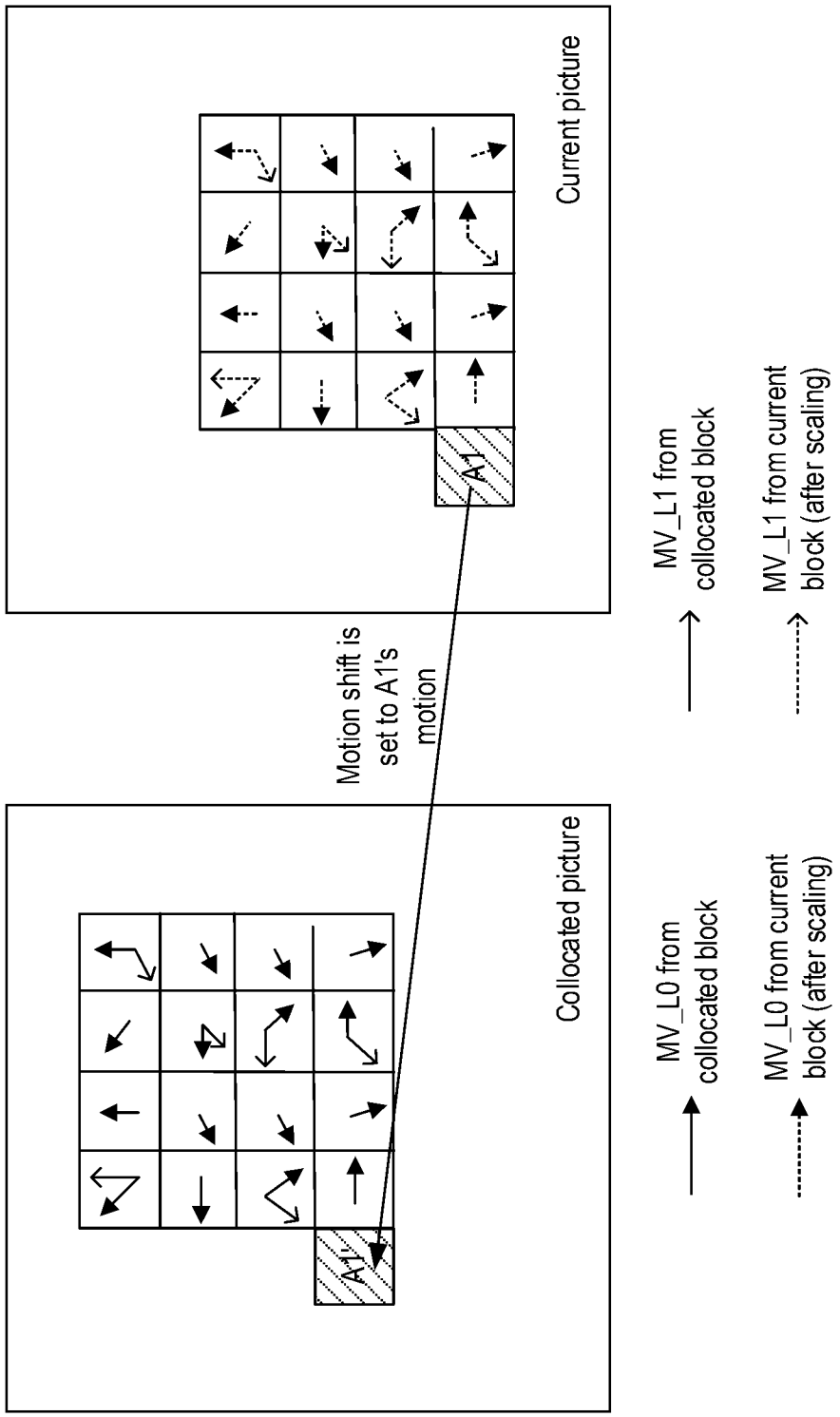
Current picture
A1
Motion shift is set to A1's motion
Collocated picture
A1'
MV_L1 from collocated block
MV_L1 from current block (after scaling)
MV_L0 from collocated block
MV_L0 from current block (after scaling)

Fig. 19B

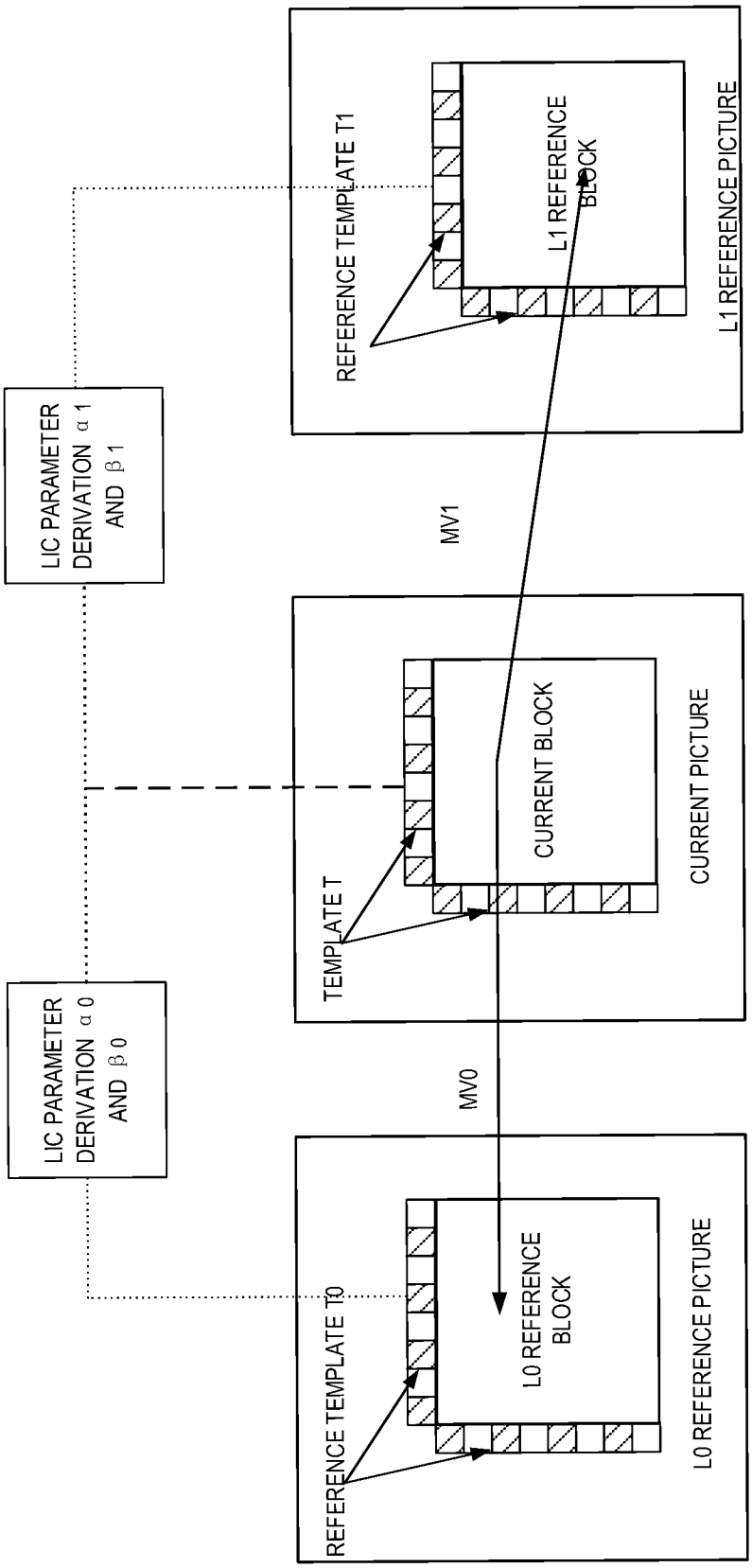
L1 REFERENCE PICTURE
REFERENCE TEMPLATE T1
L1 REFERENCE BLOCK
MV1
LIC PARAMETER DERIVATION α 1 AND β 1
CURRENT PICTURE
TEMPLATE T
CURRENT BLOCK
MV0
LIC PARAMETER DERIVATION α 0 AND β 0
L0 REFERENCE PICTURE
REFERENCE TEMPLATE T0
L0 REFERENCE BLOCK

Fig. 20

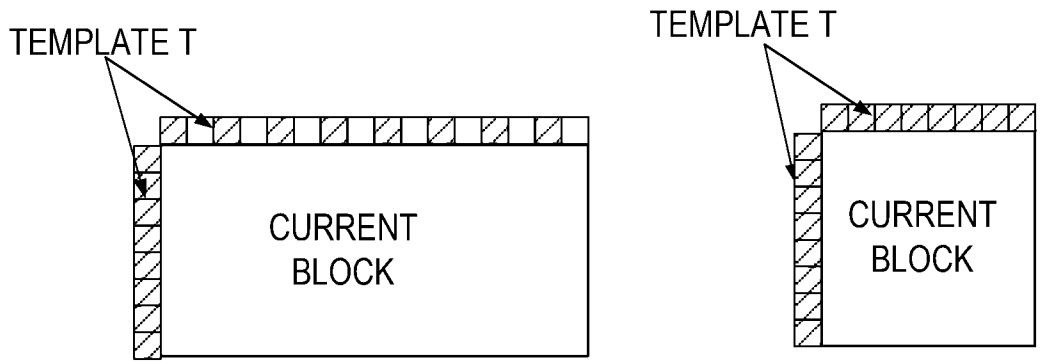
TEMPLATE T
CURRENT BLOCK
TEMPLATE T
CURRENT BLOCK

Fig. 21

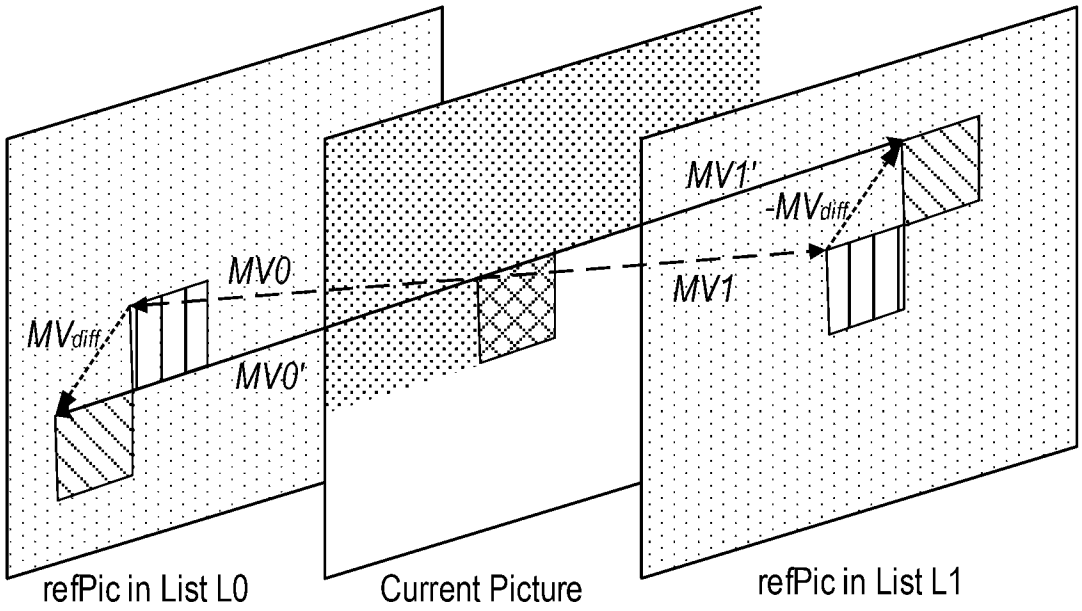
MV1'
-MVdiff
MV0
MV1
MVdiff
MV0'
refPic in List L0
Current Picture
refPic in List L1

Fig. 22

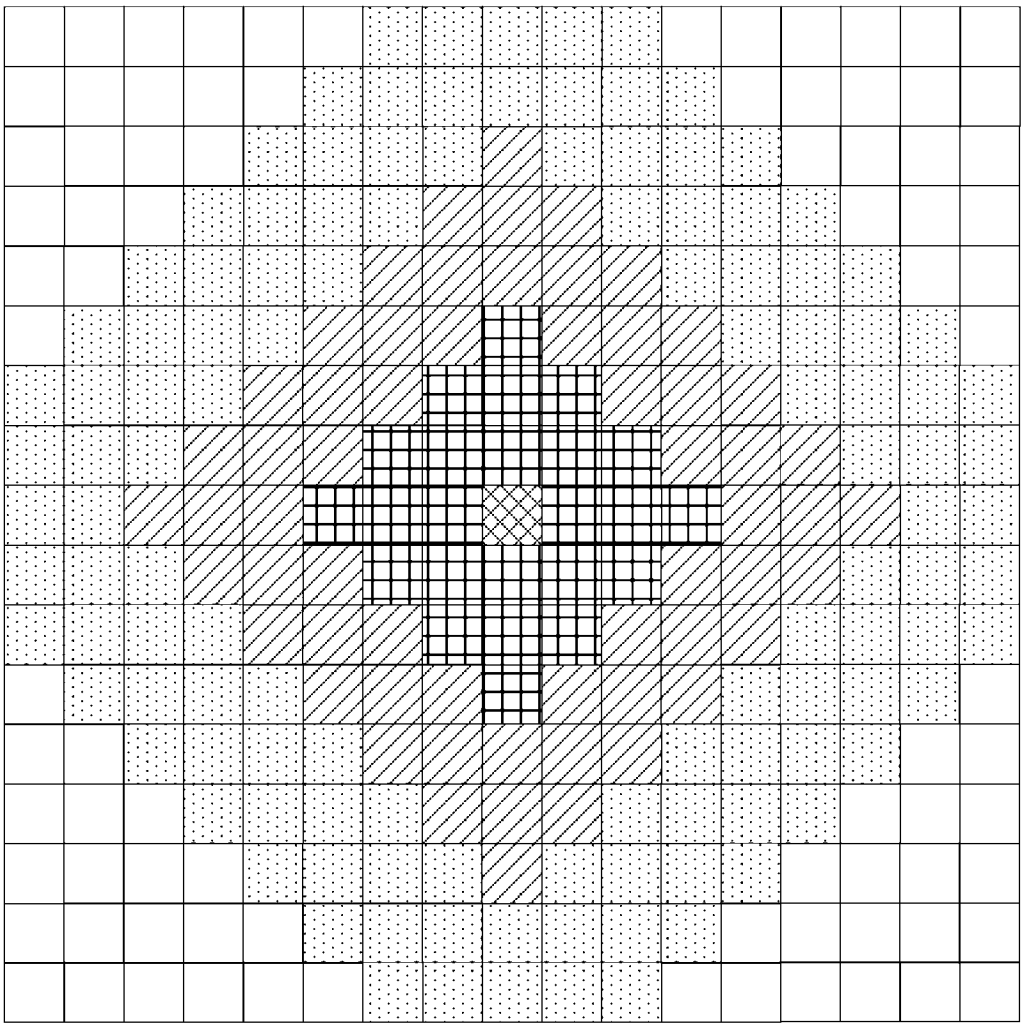

Fig. 23

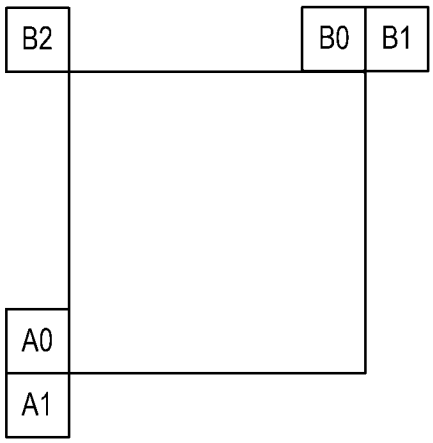

Fig. 24

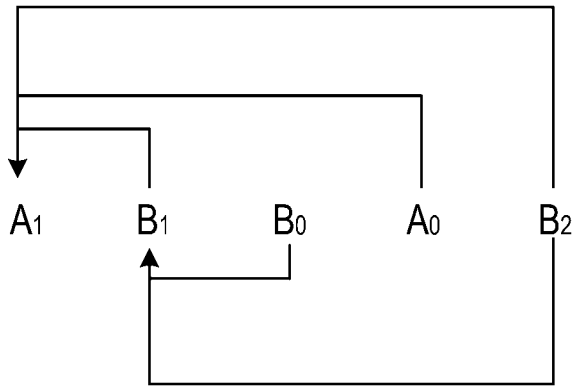

Fig. 25

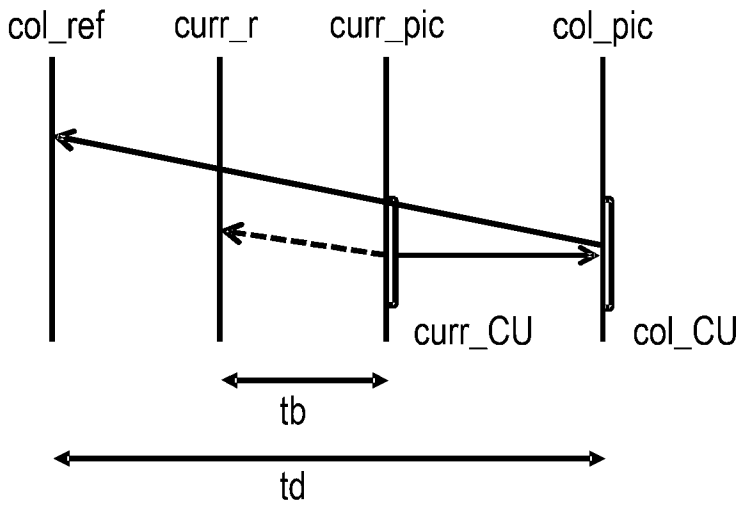

Fig. 26

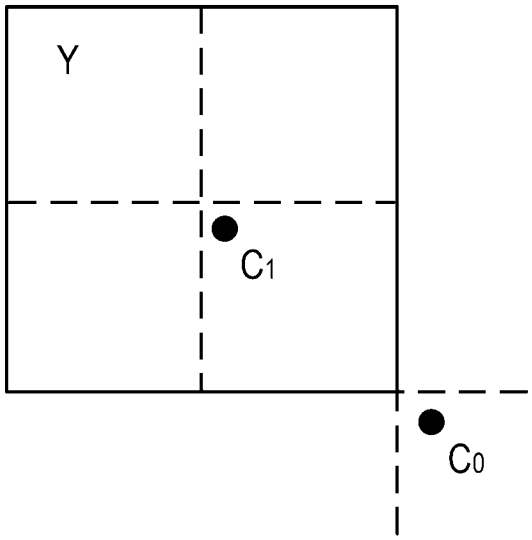

Fig. 27

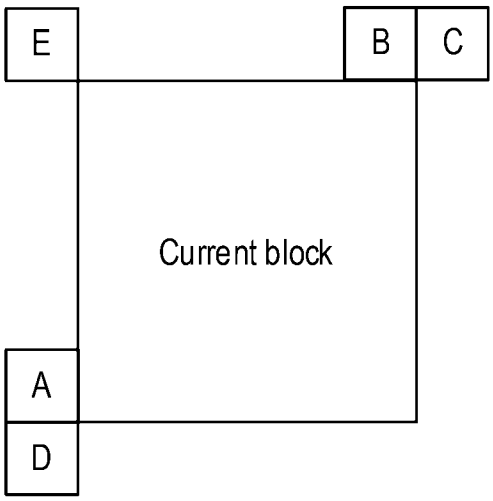

Fig. 28

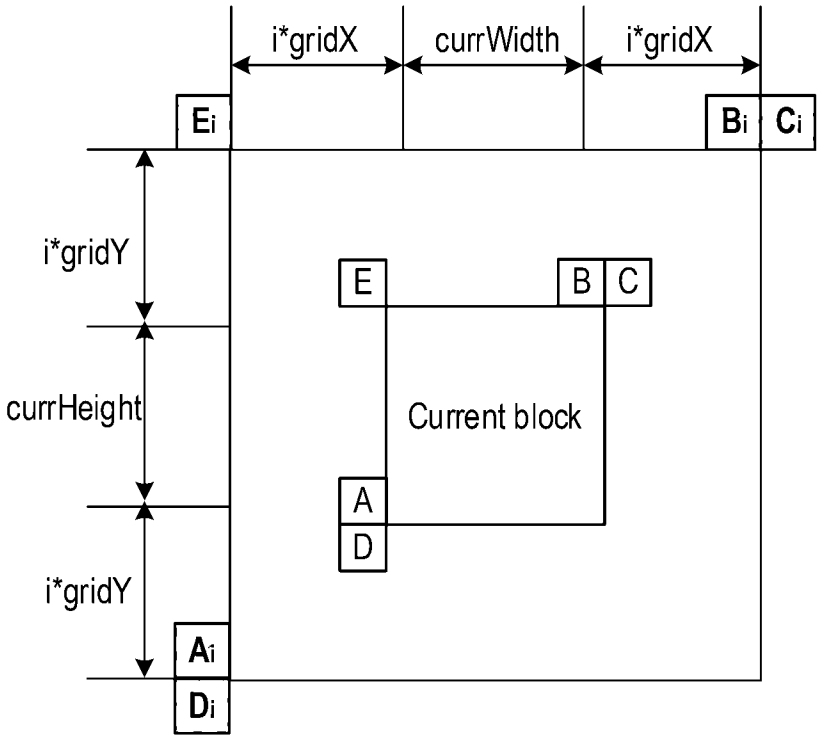
i*gridX
currWidth
i*gridX
Ei
Bi
Ci
i*gridY
E
B
C
currHeight
Current block
A
D
i*gridY
Ai
Di

Fig. 29

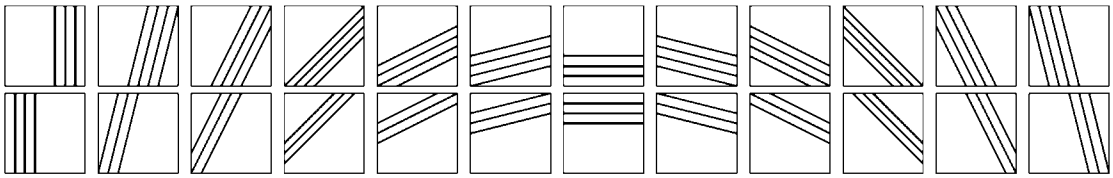

Fig. 30

Merge index	L0 MV	L1 MV
0	x	
1		x
2	x	
3		x
4	x	

Fig. 31

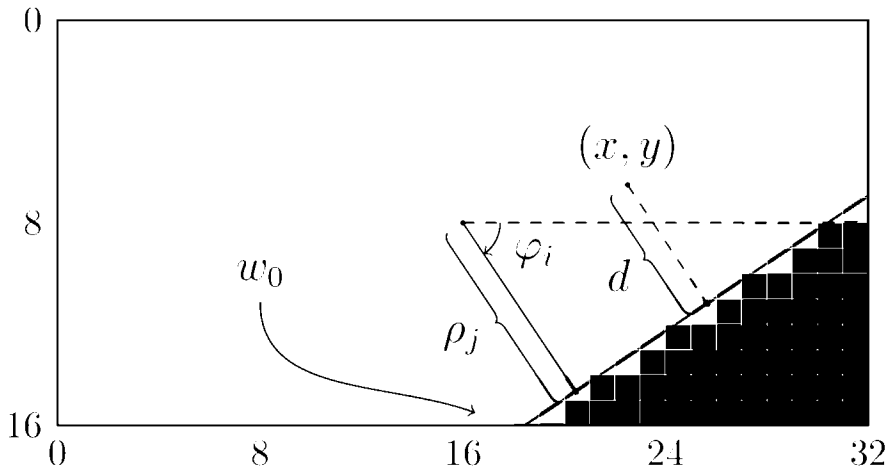

Fig. 32

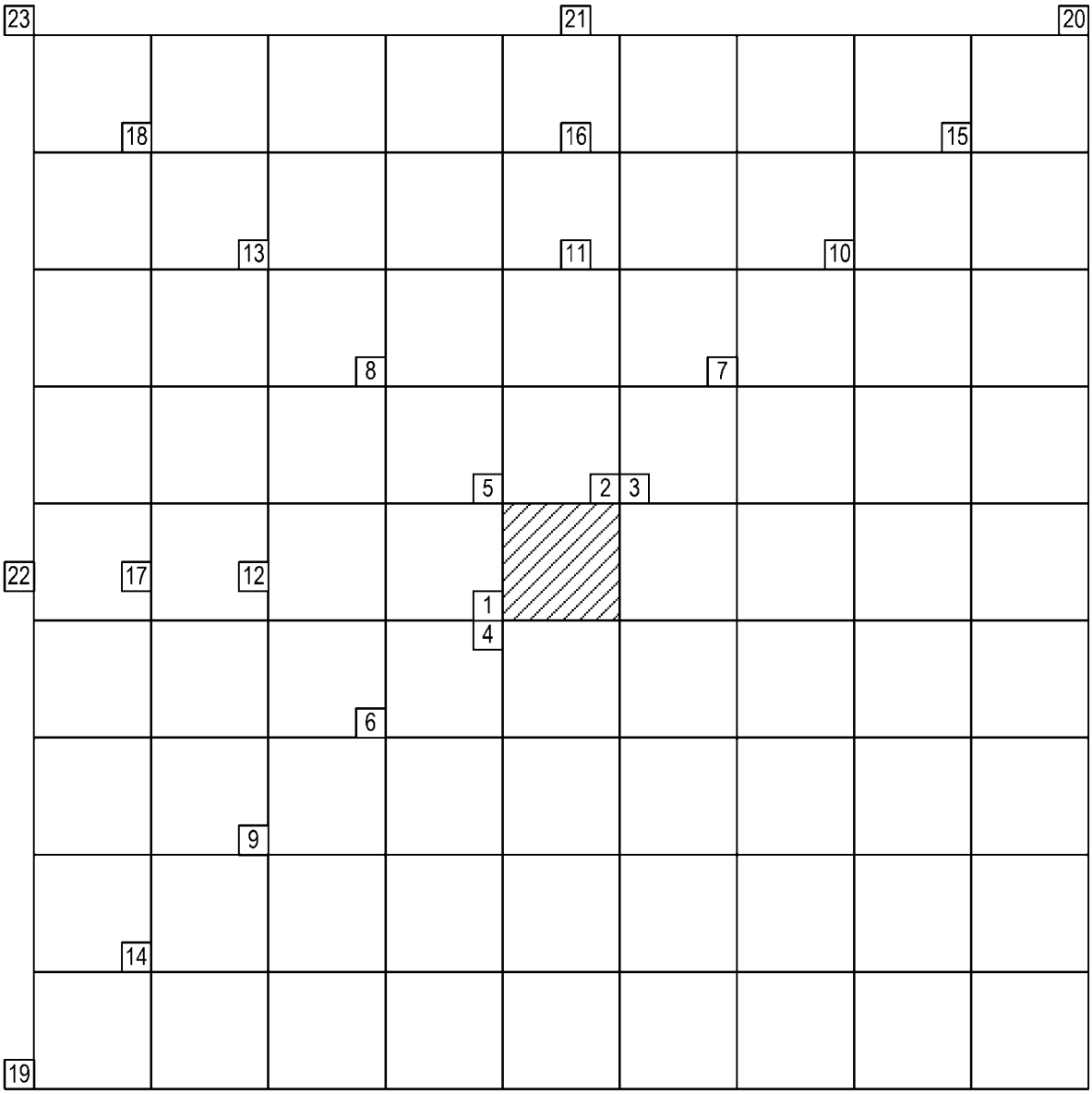

Fig. 33

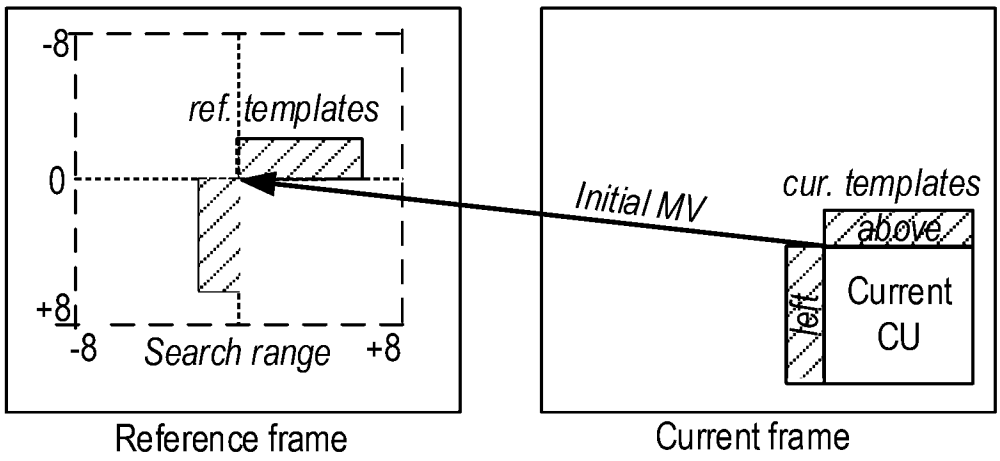
-8
ref. templates
0
+8
-8
Search range
+8
Reference frame
Initial MV
cur. templates
above
left
Current
CU
Current frame

Fig. 34

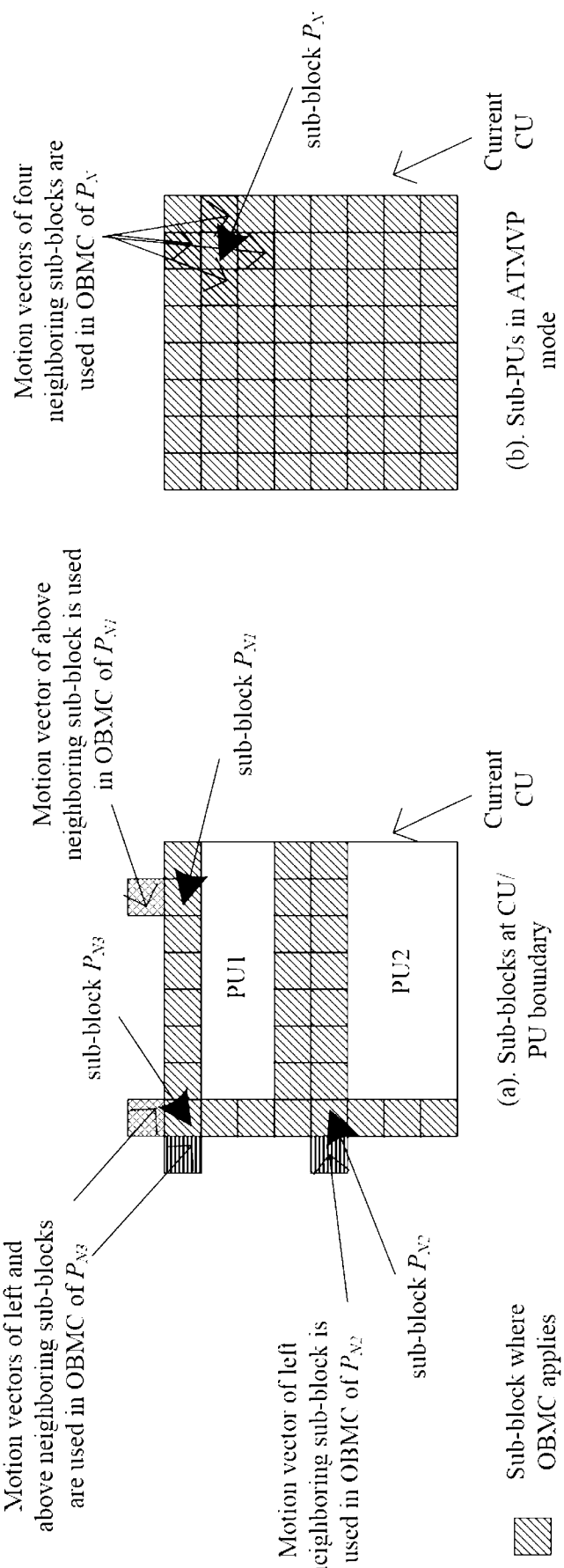
Motion vectors of four neighboring sub-blocks are used in OBMC of P_N
sub-block P_N
Current CU
(b). Sub-PUs in ATMVP mode
Motion vector of above neighboring sub-block is used in OBMC of P_{N1}
sub-block P_{N1}
sub-block P_{N3}
PU1
PU2
Current CU
(a). Sub-blocks at CU/PU boundary
Motion vectors of left and above neighboring sub-blocks are used in OBMC of P_{N3}
Motion vector of left neighboring sub-block is used in OBMC of P_{N2}
sub-block P_{N2}
Sub-block where OBMC applies

Fig. 35

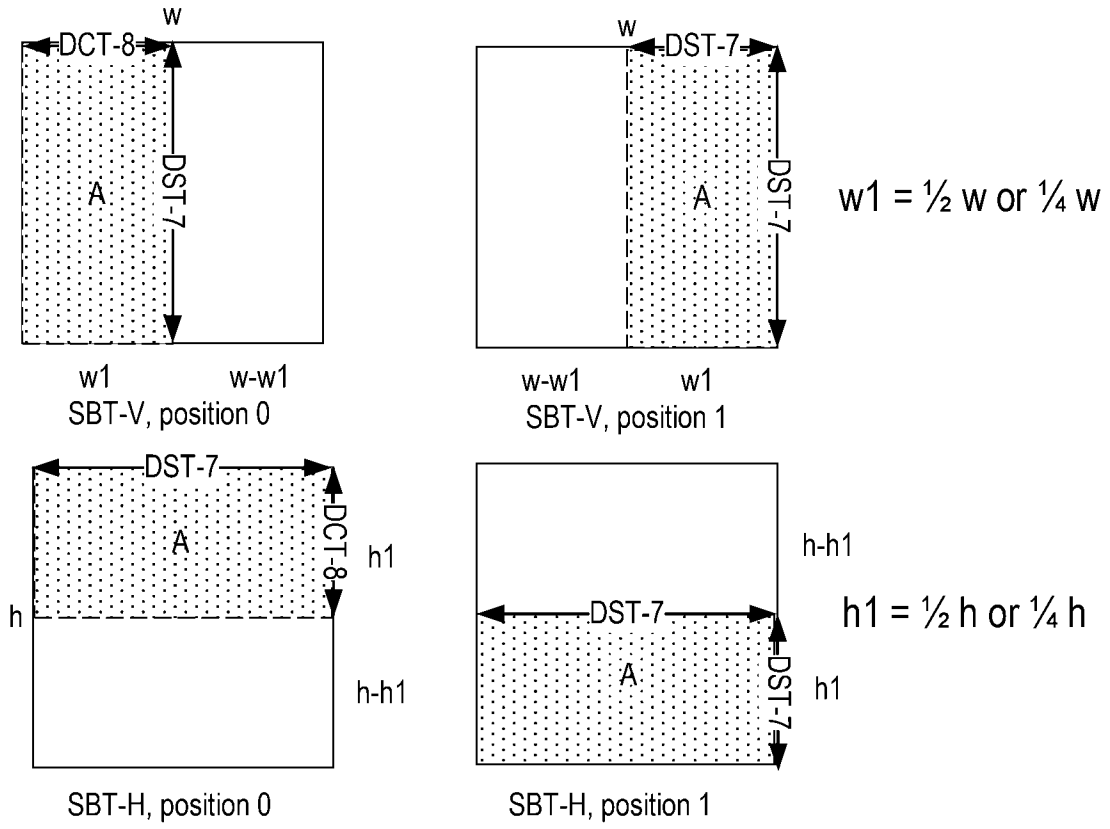
w
DCT-8
A
DST-7
w1
w-w1
SBT-V, position 0
w
DST-7
A
DST-7
w-w1
w1
SBT-V, position 1
w1 = ½ w or ¼ w
DST-7
A
DCT-8
h
h1
h-h1
SBT-H, position 0
h-h1
DST-7
A
DST-7
h1
SBT-H, position 1
h1 = ½ h or ¼ h

Fig. 36

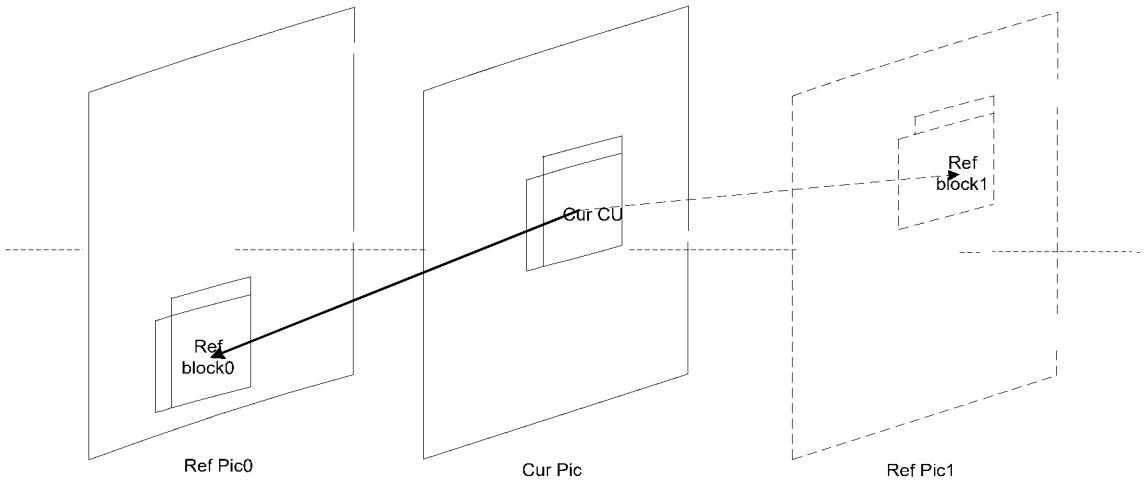
Ref block1
Cur CU
Ref block0
Ref Pic0
Cur Pic
Ref Pic1

Fig. 37

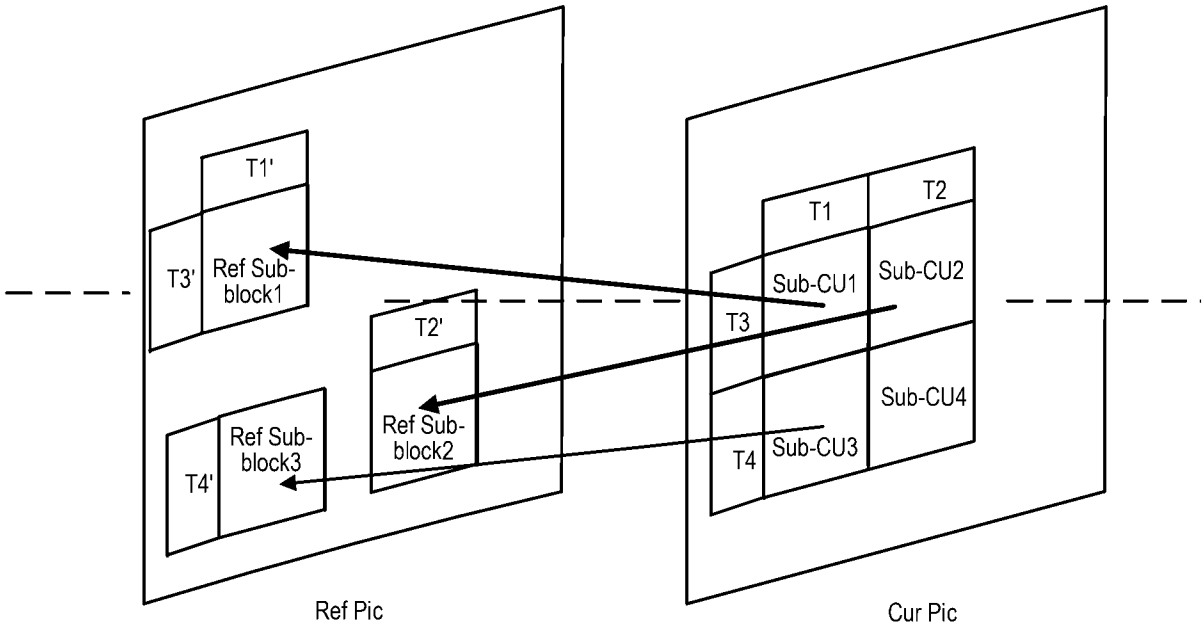
T1'
T3'
Ref Sub-block1
T2'
Ref Sub-block2
T4'
Ref Sub-block3
Ref Pic
T1
T2
Sub-CU1
Sub-CU2
T3
Sub-CU4
T4
Sub-CU3
Cur Pic

Fig. 38

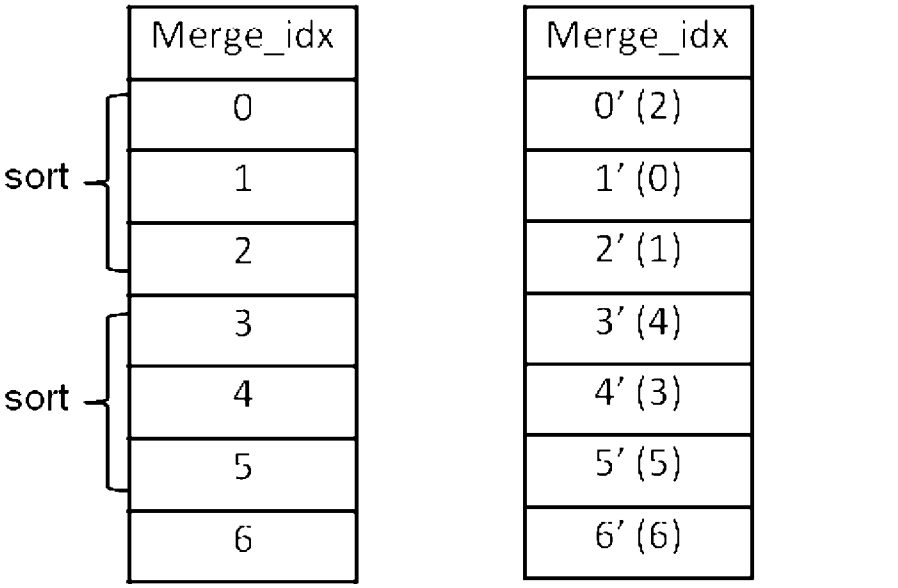
Merge_idx
0
1
2
3
4
5
6
sort
sort
Merge_idx
0' (2)
1' (0)
2' (1)
3' (4)
4' (3)
5' (5)
6' (6)
Original merge candidate list
Updated merge candidate list

Fig. 39

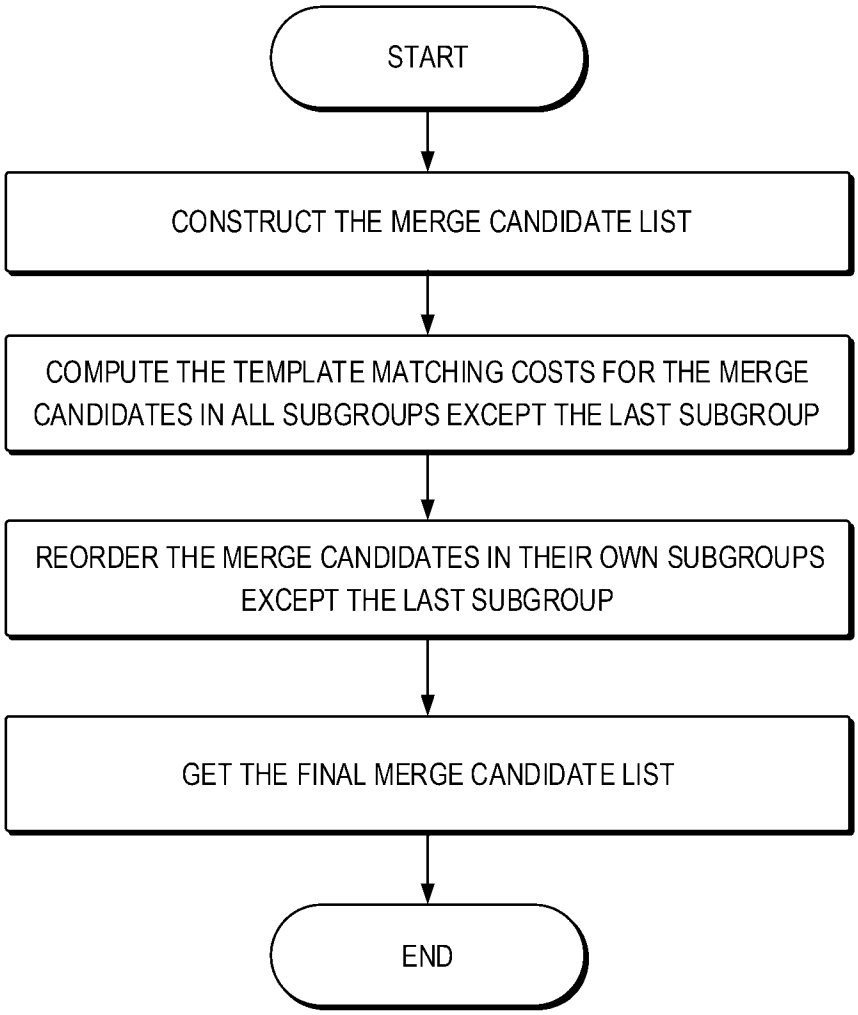
START
CONSTRUCT THE MERGE CANDIDATE LIST
COMPUTE THE TEMPLATE MATCHING COSTS FOR THE MERGE CANDIDATES IN ALL SUBGROUPS EXCEPT THE LAST SUBGROUP
REORDER THE MERGE CANDIDATES IN THEIR OWN SUBGROUPS EXCEPT THE LAST SUBGROUP
GET THE FINAL MERGE CANDIDATE LIST
END

Fig. 40

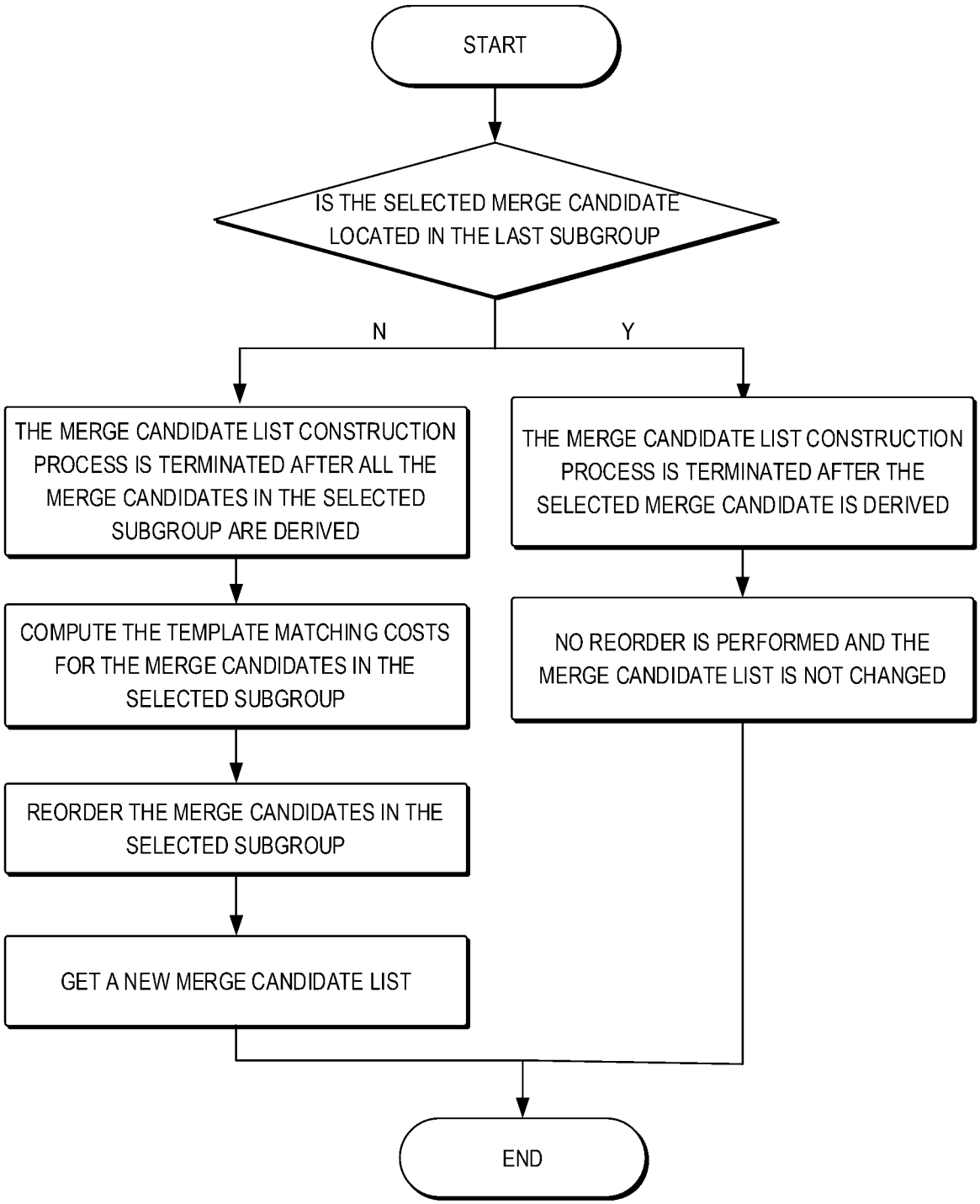
START
IS THE SELECTED MERGE CANDIDATE LOCATED IN THE LAST SUBGROUP
N
Y
THE MERGE CANDIDATE LIST CONSTRUCTION PROCESS IS TERMINATED AFTER ALL THE MERGE CANDIDATES IN THE SELECTED SUBGROUP ARE DERIVED
COMPUTE THE TEMPLATE MATCHING COSTS FOR THE MERGE CANDIDATES IN THE SELECTED SUBGROUP
REORDER THE MERGE CANDIDATES IN THE SELECTED SUBGROUP
GET A NEW MERGE CANDIDATE LIST
THE MERGE CANDIDATE LIST CONSTRUCTION PROCESS IS TERMINATED AFTER THE SELECTED MERGE CANDIDATE IS DERIVED
NO REORDER IS PERFORMED AND THE MERGE CANDIDATE LIST IS NOT CHANGED
END

Fig. 41

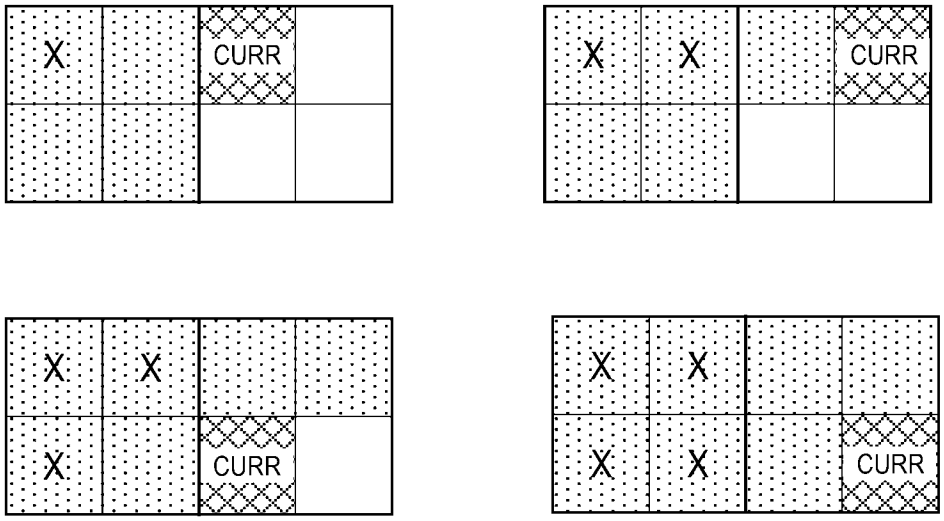

Fig. 42

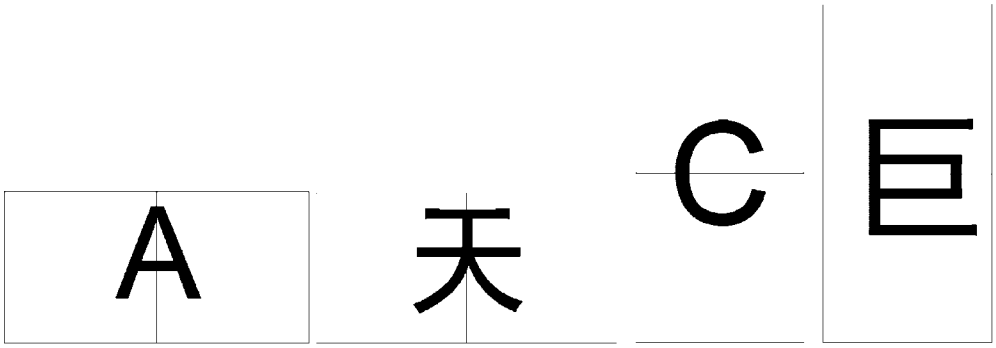

Fig. 43

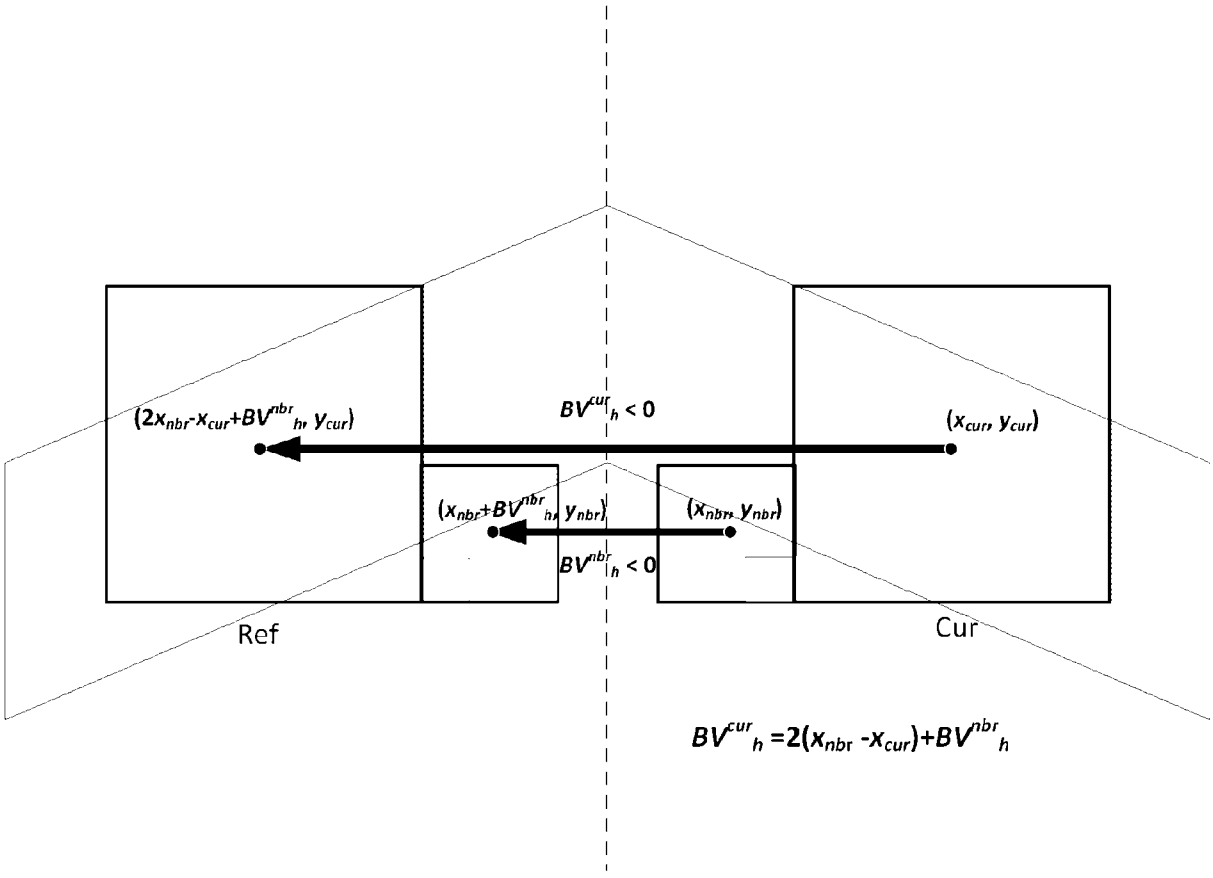
$(2x_{nbr}-x_{cur}+BV^{nbr}_h, y_{cur})$
$BV^{cur}_h < 0$
(x_{cur}, y_{cur})
$(x_{nbr}+BV^{nbr}_h, y_{nbr})$
(x_{nbr}, y_{nbr})
$BV^{nbr}_h < 0$
Ref
Cur
$BV^{cur}_h = 2(x_{nbr} - x_{cur}) + BV^{nbr}_h$

Fig. 44A

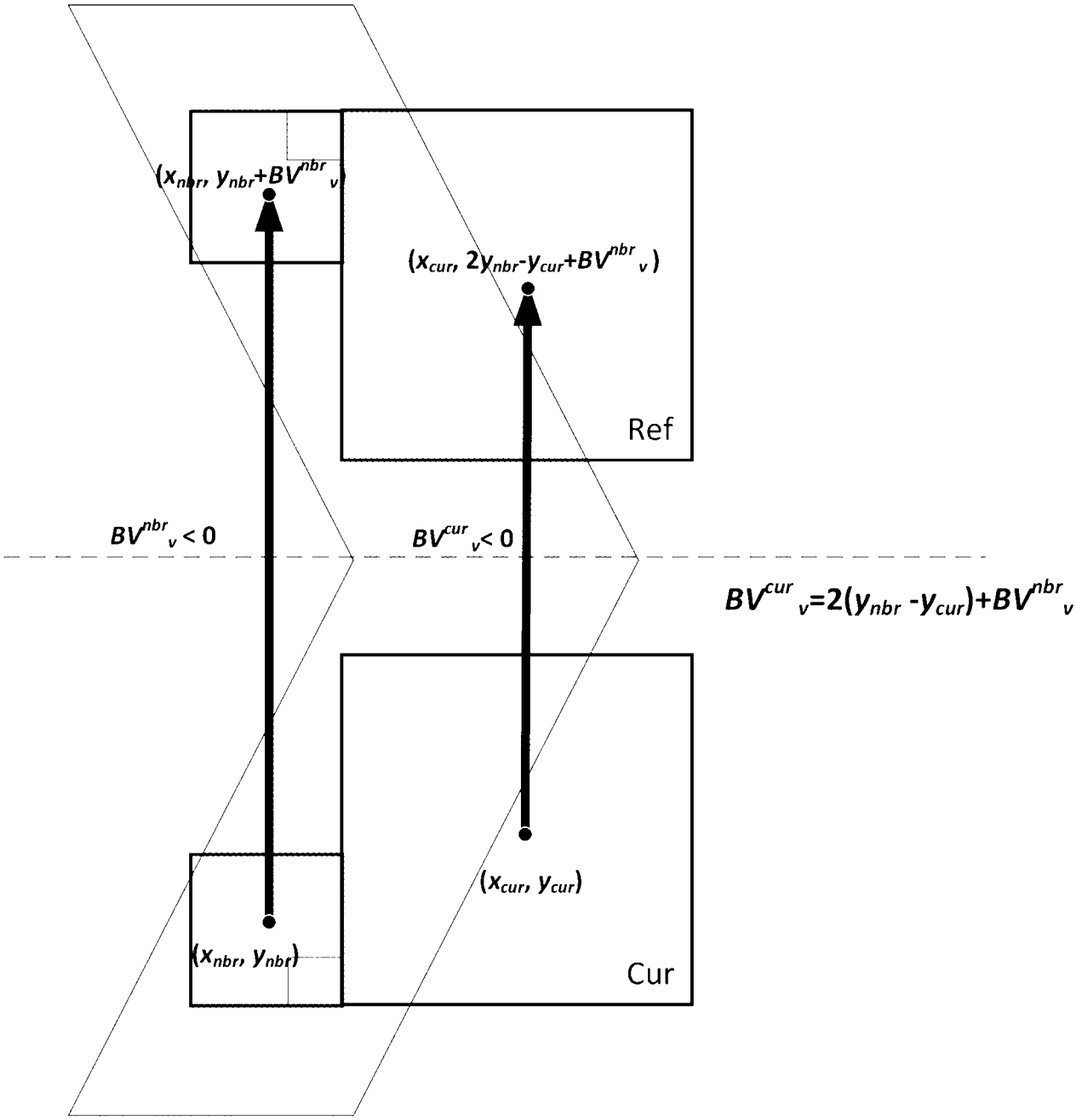
$(x_{nbr}, y_{nbr}+BV^{nbr}_{v})$
$(x_{cur}, 2y_{nbr}-y_{cur}+BV^{nbr}_{v})$
Ref
$BV^{nbr}_{v}<0$
$BV^{cur}_{v}<0$
$BV^{cur}_{v}=2(y_{nbr}-y_{cur})+BV^{nbr}_{v}$
(x_{cur}, y_{cur})
(x_{nbr}, y_{nbr})
Cur

Fig. 44B

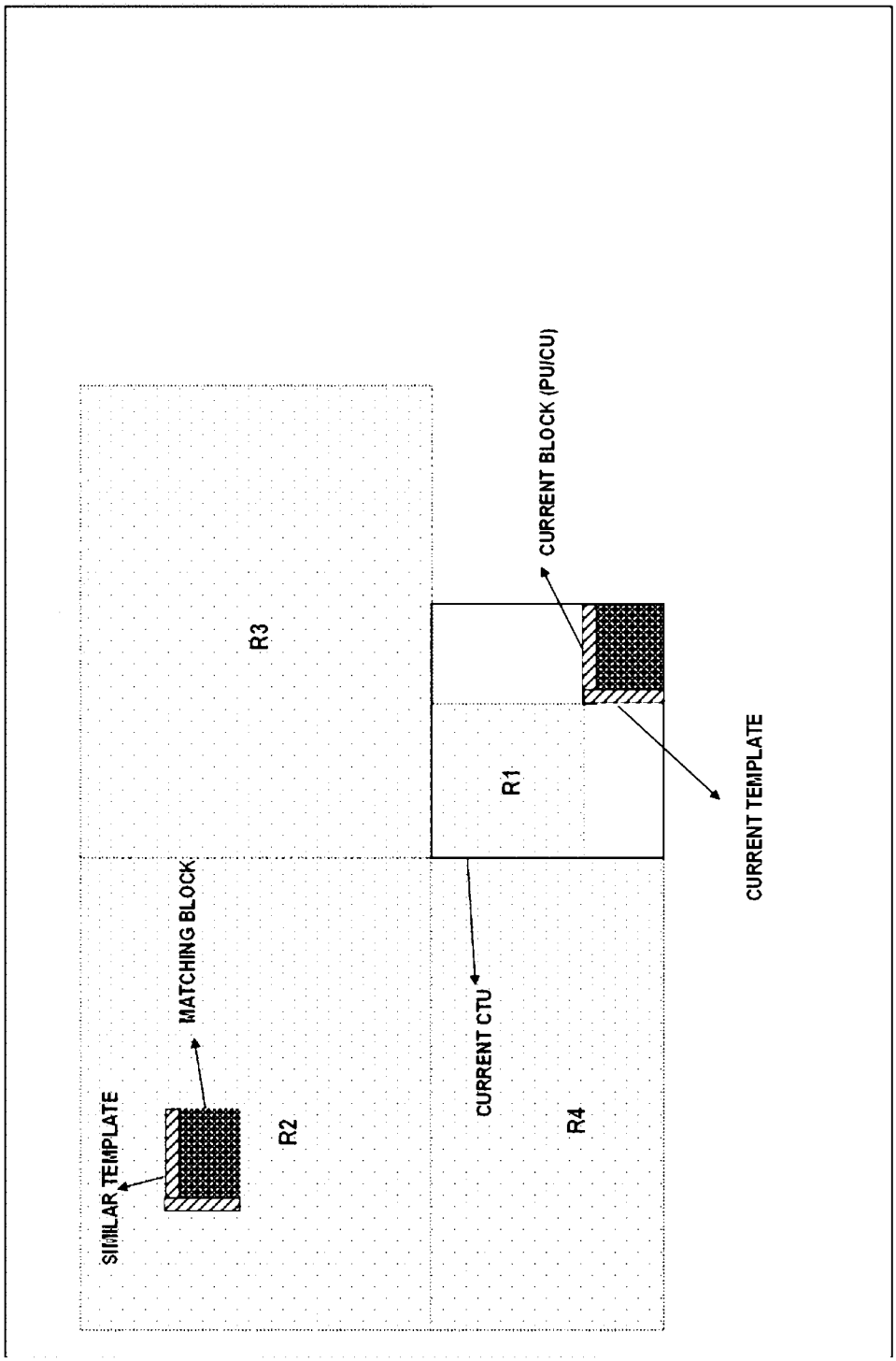
CURRENT BLOCK (PU/CU)
R3
R1
CURRENT TEMPLATE
MATCHING BLOCK
SIMILAR TEMPLATE
R2
CURRENT CTU
R4

Fig. 45

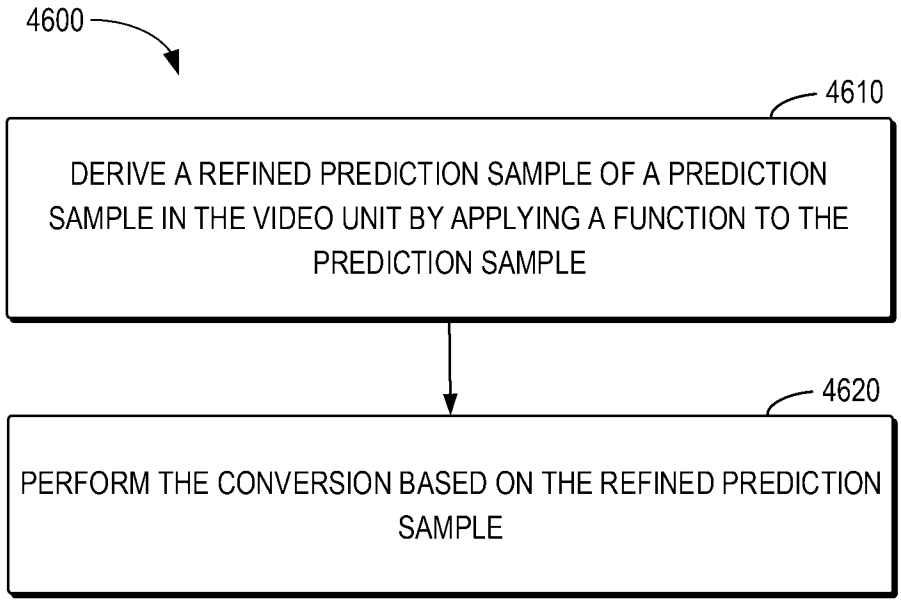
4600
4610
DERIVE A REFINED PREDICTION SAMPLE OF A PREDICTION SAMPLE IN THE VIDEO UNIT BY APPLYING A FUNCTION TO THE PREDICTION SAMPLE
4620
PERFORM THE CONVERSION BASED ON THE REFINED PREDICTION SAMPLE

Fig. 46

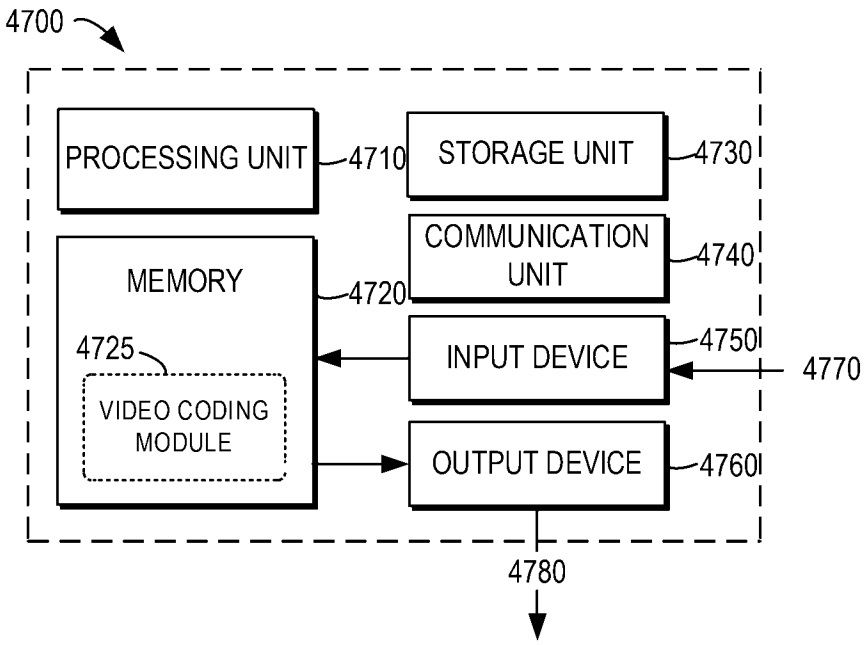
4700
PROCESSING UNIT
4710
STORAGE UNIT
4730
COMMUNICATION UNIT
4740
MEMORY
4720
4725
VIDEO CODING MODULE
INPUT DEVICE
4750
4770
OUTPUT DEVICE
4760
4780

Fig. 47

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2024/102064

A. CLASSIFICATION OF SUBJECT MATTER

H04N19/52(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)

IPC: 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)

CNTXT; CNABS; VEN; ENTXT; ENTXTC; WPABS; WPABSC; CNKI; CSDN:video, refine, prediction, sample, function, local, illumination, compensation, LIC, target, mode, coding, parameter, modify

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CN 114503561 A (DAJIA INTERNET INFORMATION TECH CO LTD) 13 May 2022 (2022-05-13) the whole document	1-79
A	CN 115918080 A (DOUYIN VISION CO LTD et. al.) 04 April 2023 (2023-04-04) the whole document	1-79
A	CN 114342378 A (DAJIA INTERNET INFORMATION TECH CO LTD) 12 April 2022 (2022-04-12) the whole document	1-79
A	US 2022109887 A1 (LG ELECTRONICS INC) 07 April 2022 (2022-04-07) the whole document	1-79

☐ 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

"D" document cited by the applicant in the international application

"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	Date of mailing of the international search report
20 September 2024	**24 September 2024**
Name and mailing address of the ISA/CN **CHINA NATIONAL INTELLECTUAL PROPERTY ADMINISTRATION** **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing 100088, China**	Authorized officer **WANG,Qian** Telephone No. **(+86) 010-62412164**

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2024/102064

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	114503561	A	13 May 2022	JP	2022541685	A	26 September 2022
				JP	7281602	B2	25 May 2023
				US	2022239943	A1	28 July 2022
				US	12069297	B2	20 August 2024
				EP	4022923	A1	06 July 2022
				EP	4022923	A4	23 November 2022
				KR	20230070534	A	23 May 2023
				KR	20230018532	A	07 February 2023
				JP	2023100979	A	19 July 2023
				WO	2021072326	A1	15 April 2021
				KR	20220046707	A	14 April 2022
				KR	102533731	B1	19 June 2023
				KR	20230070535	A	23 May 2023
				MX	2022004332	A	26 April 2022
CN	115918080	A	04 April 2023	WO	2021249375	A1	16 December 2021
				US	2023107138	A1	06 April 2023
				US	11956448	B2	09 April 2024
				US	2024129489	A1	18 April 2024
CN	114342378	A	12 April 2022	WO	2021007557	A1	14 January 2021
				JP	2023089253	A	27 June 2023
				JP	2023164968	A	14 November 2023
				JP	2022531629	A	07 July 2022
				JP	7313533	B2	24 July 2023
				KR	20220140047	A	17 October 2022
				US	2022132138	A1	28 April 2022
				EP	3991431	A1	04 May 2022
				EP	3991431	A4	28 September 2022
				MX	2022000395	A	06 April 2022
				KR	20220025205	A	03 March 2022
				KR	102453806	B1	11 October 2022
				JP	2023159292	A	31 October 2023
US	2022109887	A1	07 April 2022	US	11438630	B2	06 September 2022
				JP	2024114823	A	23 August 2024
				JP	2022538026	A	31 August 2022
				US	2022345749	A1	27 October 2022
				US	11962810	B2	16 April 2024
				EP	3979645	A1	06 April 2022
				EP	3979645	A4	10 August 2022
				KR	20210148366	A	07 December 2021
				WO	2020256487	A1	24 December 2020