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

(19) **World Intellectual Property Organization**
International Bureau

(43) **International Publication Date**
11 January 2024 (11.01.2024)

WIPO | PCT

(10) **International Publication Number**
WO 2024/008021 A1

(51) **International Patent Classification:**
H04N 19/176 (2014.01) *H04N 19/52* (2014.01)
H04N 19/51 (2014.01)

(21) **International Application Number:** PCT/CN2023/105229

(22) **International Filing Date:** 30 June 2023 (30.06.2023)

(25) **Filing Language:** English

(26) **Publication Language:** English

(30) **Priority Data:**
PCT/CN2022/103765 04 July 2022 (04.07.2022) CN

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

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

(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,

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

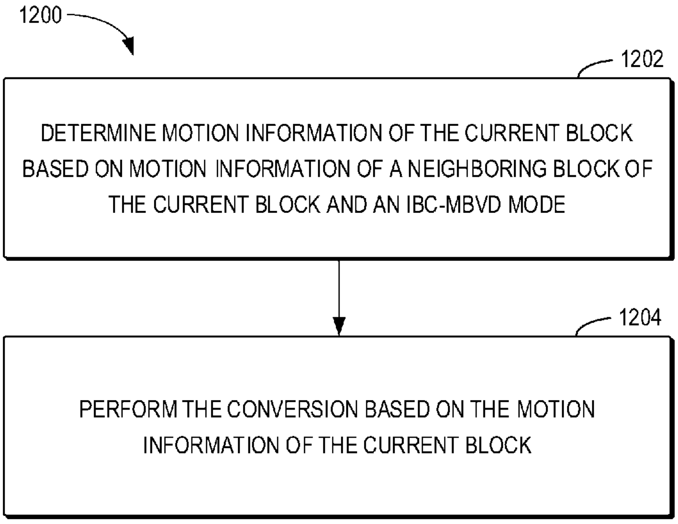

Fig. 12

(57) **Abstract:** Embodiments of the present disclosure provide a solution for video processing. A method for video processing is proposed. The method comprises: determining, for a conversion between a current block of a video and a bitstream of the video, motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy merge mode with block vector difference (IBC-MBVD) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and performing the conversion based on the motion information of the current block.

[Continued on next page]

RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, 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 an interaction of reconstruction-reordered intra block copy (RRIBC) and other IBC-based modes.

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 quality and 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: determining, for a conversion between a current block of a video and a bitstream of the video, motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy merge mode with block vector difference (IBC-MBVD) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and performing the conversion based on the motion information of the current block.

[0005] According to the method in accordance with the first aspect of the present disclosure, the motion information of the current block is determined based on motion information of an RRIBC-coded neighboring block and an IBC-MBVD mode. Compared with the conventional solution, the proposed method can better support the interaction between RRIBC mode and IBC-MBVD mode, and thus can advantageously improve the coding efficiency and coding quality.

[0006] In a second aspect, another method for video processing is proposed. The method comprises: determining, for a conversion between a current block of a video and a bitstream of the video, the motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy template matching merge (IBC-TM-MERGE) mode, the current block and the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and performing the conversion based on the motion information of the current block.

[0007] According to the method in accordance with the second aspect of the present disclosure, motion information of the RRIBC-coded current block is determined based on motion information of an RRIBC-coded neighboring block and an IBC-TM-MERGE mode. Compared with the conventional solution, the proposed method can better support the interaction between RRIBC mode and IBC-TM-MERGE mode, and thus can advantageously improve the coding efficiency and coding quality.

[0008] In a third aspect, another method for video processing is proposed. The method comprises: determining, for a conversion between a current block of a video and a bitstream of the video, a motion candidate for the current block based on motion information of a neighboring block of the current block and an intra block copy advanced motion vector prediction (IBC-AMVP) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; determining motion information of the current block based on the motion candidate; and performing the conversion based on the motion information of the current block.

[0009] According to the method in accordance with the third aspect of the present disclosure, the motion information of the current block is determined based on motion information of an RRIBC-coded neighboring block and an IBC-AMVP mode. Compared with the conventional solution, the proposed method can better support the interaction between RRIBC mode and IBC-AMVP mode, and thus can advantageously improve the coding efficiency and coding quality.

[0010] In a fourth 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.

[0011] In a fifth 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.

[0012] In a sixth 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: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-MBVD mode, the neighboring block being coded with an RRIBC mode; and generating the bitstream based on the motion information of the current block.

[0013] In a seventh aspect, a method for storing a bitstream of a video is proposed. The method comprises: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-MBVD mode, the neighboring block being coded with an RRIBC mode; generating the bitstream based on the motion information of the current block; storing the bitstream in a non-transitory computer-readable recording medium.

[0014] In an eighth 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: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode, the current block and the neighboring block being coded with an RRIBC mode; and generating the bitstream based on the motion information of the current block.

[0015] In a ninth aspect, a method for storing a bitstream of a video is proposed. The method comprises: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode, the current block and the neighboring block being coded with an RRIBC mode; generating the bitstream based on the motion information of the current block; and storing the bitstream in a non-transitory computer-readable recording medium.

[0016] In a tenth 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: determining a motion candidate for a current block of the video based on motion information of a neighboring block of the current block and an IBC-AMVP mode, the neighboring block being coded with an RRIBC mode; determining motion information of the current block based on the motion candidate; and generating the bitstream based on the motion information of the current block.

[0017] In an eleventh aspect, a method for storing a bitstream of a video is proposed. The method comprises: determining a motion candidate for a current block of the video based on motion information of a neighboring block of the current block and an IBC-AMVP mode, the neighboring block being coded with an RRIBC mode; determining motion information of the current block based on the motion candidate; generating the bitstream based on the motion information of the current block; and storing the bitstream in a non-transitory computer-readable recording medium.

[0018] 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

[0019] 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.

[0020] Fig. 1 illustrates a block diagram that illustrates an example video coding system, in accordance with some embodiments of the present disclosure;

[0021] Fig. 2 illustrates a block diagram that illustrates a first example video encoder, in accordance with some embodiments of the present disclosure;

[0022] Fig. 3 illustrates a block diagram that illustrates an example video decoder, in accordance with some embodiments of the present disclosure;

[0023] Fig. 4 illustrates current coding tree unit (CTU) processing order and its available reference samples in current and left CTU;

[0024] Fig. 5 illustrates residual coding passes for transform skip blocks;

[0025] Fig. 6 illustrates an example of a block coded in palette mode;

[0026] Fig. 7 illustrates subblock-based index map scanning for palette;

[0027] Fig. 8 illustrates a decoding flowchart with adaptive color transform (ACT);

[0028] Fig. 9 illustrates an intra template matching search area used;

[0029] Fig. 10 illustrates an intra block copy (IBC) reference region depending on current coding unit (CU) position;

[0030] Fig. 11A illustrates an example of block vector (BV) adjustment for a horizontal flip;

[0031] Fig. 11B illustrates an example of BV adjustment for a vertical flip;

[0032] Fig. 12 illustrates a flowchart of a method for video processing in accordance with embodiments of the present disclosure;

[0033] Fig. 13 illustrates a flowchart of another method for video processing in accordance with embodiments of the present disclosure;

[0034] Fig. 14 illustrates a flowchart of a further method for video processing in accordance with embodiments of the present disclosure;

[0035] Fig. 15 illustrates a block diagram of a computing device in which various embodiments of the present disclosure can be implemented.

[0036] Throughout the drawings, the same or similar reference numerals usually refer to the same or similar elements.

DETAILED DESCRIPTION

[0037] 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.

[0038] 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.

[0039] 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.

[0040] 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.

[0041] 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

[0042] 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.

[0043] 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.

[0044] 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.

[0045] 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.

[0046] 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.

[0047] 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.

[0048] 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.

[0049] In some embodiments, the video encoder 200 may include a partition unit 201, a prediction 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.

[0050] In other examples, the video encoder 200 may include more, fewer, or different functional components. In an example, the prediction unit 202 may include an intra block copy (IBC) unit. The IBC unit may perform prediction in an IBC mode in which at least one reference picture is a picture where the current video block is located.

[0051] 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.

[0052] 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.

[0053] 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 prediction (CIIP) mode in which the prediction is based on an inter prediction signal and an intra prediction 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-prediction.

[0054] 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.

[0055] 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.

[0056] 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.

[0057] 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.

[0058] 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.

[0059] 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.

[0060] 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.

[0061] 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 prediction (AMVP) and merge mode signaling.

[0062] 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.

[0063] 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.

[0064] 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.

[0065] 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.

[0066] 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.

[0067] 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 prediction unit 202 to produce a reconstructed video block associated with the current video block for storage in the buffer 213.

[0068] After the reconstruction unit 212 reconstructs the video block, loop filtering operation may be performed to reduce video blocking artifacts in the video block.

[0069] 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.

[0070] 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.

[0071] 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.

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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.

[0076] 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.

[0077] 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.

[0078] The reconstruction unit 306 may obtain the decoded blocks, e.g., by summing the residual blocks with the corresponding prediction blocks generated by the motion compensation unit 302 or intra-prediction unit 303. If desired, a deblocking filter may also be applied to filter the decoded blocks in order to remove blockiness artifacts. The decoded video blocks are then stored in the buffer 307, which provides reference blocks for subsequent motion compensation/intra prediction and also produces decoded video for presentation on a display device.

[0079] 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

This disclosure is related to video coding technologies. Specifically, it is about the interaction of RRIBC and other coding tools in image/video coding. It may be applied to the existing video coding standard like HEVC, VVC, and etc. It may be also applicable to future video coding standards or video codec.

2 Introduction

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, the Joint Video Exploration Team (JVET) was founded by VCEG and MPEG jointly in 2015. The JVET meeting is concurrently held once every quarter, and the new video coding standard was officially named as Versatile Video Coding (VVC) in the April 2018 JVET meeting, and the first version of VVC test model (VTM) was released at that time. The VVC working draft and test model VTM are then updated after every meeting. The VVC project achieved technical completion (FDIS) at the July 2020 meeting.

In January 2021, JVET established an Exploration Experiment (EE), targeting at enhanced compression efficiency beyond VVC capability with novel traditional algorithms. Soon later, ECM was built as the common software base for longer-term exploration work towards the next generation video coding standard.

2.1 Existing screen content coding tools

2.1.1 Intra block copy (IBC)

Intra block copy (IBC) is a tool adopted in HEVC extensions on SCC. It is well known that it significantly improves the coding efficiency of screen content materials. Since IBC mode is implemented as a block level coding mode, block matching (BM) is performed at the encoder to find the optimal block vector (or motion vector) for each CU. Here, a block vector is used to indicate the displacement from the current block to a reference block, which is already reconstructed inside the current picture. The luma block vector of an IBC-coded CU is in integer precision. The chroma block vector rounds to integer precision as well. When combined with AMVR, the IBC mode can switch between 1-pel and 4-pel motion vector precisions. An IBC-coded CU is treated as the third prediction mode other than intra or inter prediction modes. The IBC mode is applicable to the CUs with both width and height smaller than or equal to 64 luma samples.

At the encoder side, hash-based motion estimation is performed for IBC. The encoder performs RD check for blocks with either width or height no larger than 16 luma samples. For non-merge mode, the block vector search is performed using hash-based search first. If hash search does not return valid candidate, block matching based local search will be performed.

In the hash-based search, hash key matching (32-bit CRC) between the current block and a reference block is extended to all allowed block sizes. The hash key calculation for every position in the current picture is based on 4x4 subblocks. 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 subblocks match the hash keys in the corresponding reference locations. If hash keys of multiple reference blocks are found to match that of the current block, the block vector costs of each matched reference are calculated and the one with the minimum cost is selected.

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

At CU level, IBC mode is signalled with a flag and it can be signaled 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 neighboring 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 neighbor and one from above neighbor (if IBC coded). When either neighbor is not available, a default block vector will be used as a predictor. A flag is signaled to indicate the block vector predictor index.

2.1.1.1 IBC reference region

To reduce memory consumption and decoder complexity, the IBC in VVC allows only the reconstructed portion of the predefined area including the region of current CTU and some region of the left CTU. Fig. 4 illustrates the reference region of IBC Mode, where each block represents 64x64 luma sample unit.

Depending on the location of the current coding CU location within the current CTU, the following applies:

- If current block falls into the top-left 64x64 block of the current CTU, then in addition to the already reconstructed samples in the current CTU, it can also refer to the reference samples in the bottom-right 64x64 blocks of the left CTU, using CPR mode. The current block can also refer to the reference samples in the bottom-left 64x64 block of the left

CTU and the reference samples in the top-right 64x64 block of the left CTU, using CPR mode.

- If current block falls into the top-right 64x64 block of the current CTU, then in addition to the already reconstructed samples in the current CTU, if luma location (0, 64) relative to the current CTU has not yet been reconstructed, the current block can also refer to the reference samples in the bottom-left 64x64 block and bottom-right 64x64 block of the left CTU, using CPR mode; otherwise, the current block can also refer to reference samples in bottom-right 64x64 block of the left CTU.
- If current block falls into the bottom-left 64x64 block of the current CTU, then in addition to the already reconstructed samples in the current CTU, if luma location (64, 0) relative to the current CTU has not yet been reconstructed, the current block can also refer to the reference samples in the top-right 64x64 block and bottom-right 64x64 block of the left CTU, using CPR mode. Otherwise, the current block can also refer to the reference samples in the bottom-right 64x64 block of the left CTU, using CPR mode.
- If current block falls into the bottom-right 64x64 block of the current CTU, it can only refer to the already reconstructed samples in the current CTU, using CPR mode.

This restriction allows the IBC mode to be implemented using local on-chip memory for hardware implementations.

2.1.1.2 IBC interaction with other coding tools

The interaction between IBC mode and other inter coding tools in VVC, such as pairwise merge candidate, history based motion vector predictor (HMVP), combined intra/inter prediction mode (CIIP), merge mode with motion vector difference (MMVD), and geometric partitioning mode (GPM) are as follows:

- IBC can be used with pairwise merge candidate and HMVP. A new pairwise IBC merge candidate can be generated by averaging two IBC merge candidates. For HMVP, IBC motion is inserted into history buffer for future referencing.
- IBC cannot be used in combination with the following inter tools: affine motion, CIIP, MMVD, and GPM.
- IBC is not allowed for the chroma coding blocks when DUAL_TREE partition is used.

Unlike in the HEVC screen content coding extension, the current picture is no longer included as one of the reference pictures in the reference picture list 0 for IBC prediction. The derivation process of motion vectors for IBC mode excludes all neighboring blocks in inter mode and vice versa. The following IBC design aspects are applied:

- IBC shares the same process as in regular MV merge including with pairwise merge candidate and history based motion predictor, but disallows TMVP and zero vector because they are invalid for IBC mode.
- Separate HMVP buffer (5 candidates each) is used for conventional MV and IBC.
- Block vector constraints are implemented in the form of bitstream conformance constraint, the encoder needs to ensure that no invalid vectors are present in the bitsream, and merge shall not be used if the merge candidate is invalid (out of range or 0). Such bitstream conformance constraint is expressed in terms of a virtual buffer as described below.
- For deblocking, IBC is handled as inter mode.
- If the current block is coded using IBC prediction mode, AMVR does not use quarter-pel; instead, AMVR is signaled to only indicate whether MV is inter-pel or 4 integer-pel.
- The number of IBC merge candidates can be signalled in the slice header separately from the numbers of regular, subblock, and geometric merge candidates.

A virtual buffer concept is used to describe the allowable reference region for IBC prediction mode and valid block vectors. Denote CTU size as ctbSize, the virtual buffer, ibcBuf, has width being wIbcBuf = 128x128/ctbSize and height hIbcBuf = ctbSize. For example, for a CTU size of 128x128, the size of ibcBuf is also 128x128; for a CTU size of 64x64, the size of ibcBuf is 256x64; and a CTU size of 32x32, the size of ibcBuf is 512x32.

The size of a VPDU is min(ctbSize, 64) in each dimension, $W_v = \min(\text{ctbSize}, 64)$.

The virtual IBC buffer, ibcBuf is maintained as follows.

- At the beginning of decoding each CTU row, refresh the whole ibcBuf with an invalid value −1.
- At the beginning of decoding a VPDU (xVPDU, yVPDU) relative to the top-left corner of the picture, set the ibcBuf[x][y] = −1, with x = xVPDU%wIbcBuf, …, xVPDU% wIbcBuf + W_v − 1; y = yVPDU%ctbSize, …, yVPDU%ctbSize + W_v − 1.
- After decoding a CU contains (x, y) relative to the top-left corner of the picture, set

  ibcBuf[x % wIbcBuf][y % ctbSize] = recSample[x][y]

For a block covering the coordinates (x, y), if the following is true for a block vector *bv* =

(bv[0], bv[1]), then it is valid; otherwise, it is not valid:

ibcBuf[(x + bv[0])% wIbcBuf] [(y + bv[1]) % ctbSize] shall not be equal to −1.

2.1.2 Block differential pulse coded modulation (BDPCM)

VVC supports block differential pulse coded modulation (BDPCM) for screen content coding. At the sequence level, a BDPCM enable flag is signalled in the SPS; this flag is signalled only if the transform skip mode (described in the next section) is enabled in the SPS.

When BDPCM is enabled, a flag is transmitted at the CU level if the CU size is smaller than or equal to MaxTsSize by MaxTsSize in terms of luma samples and if the CU is intra coded, where MaxTsSize is the maximum block size for which the transform skip mode is allowed. This flag indicates whether regular intra coding or BDPCM is used. If BDPCM is used, a BDPCM prediction direction flag is transmitted to indicate whether the prediction is horizontal or vertical. Then, the block is predicted using the regular horizontal or vertical intra prediction process with unfiltered reference samples. The residual is quantized and the difference between each quantized residual and its predictor, i.e. the previously coded residual of the horizontal or vertical (depending on the BDPCM prediction direction) neighbouring position, is coded.

For a block of size M (height) × N (width), let $r_{i,j},\ 0 \le i \le M-1,\ 0 \le j \le N-1$ be the prediction residual. Let $Q(r_{i,j}),\ 0 \le i \le M-1,\ 0 \le j \le N-1$ denote the quantized version of the residual $r_{i,j}$. BDPCM is applied to the quantized residual values, resulting in a modified M ×N array $\tilde{R}$ with elements $\tilde{r}_{i,j}$, where $\tilde{r}_{i,j}$ is predicted from its neighboring quantized residual value. For vertical BDPCM prediction mode, for $0 \le j \le (N-1)$, the following is used to derive $\tilde{r}_{i,j}$:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0 \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1) \end{cases} \tag{2-1}$$

For horizontal BDPCM prediction mode, for $0 \le i \le (M-1)$, the following is used to derive $\tilde{r}_{i,j}$:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 1 \le j \le (N-1) \end{cases} \tag{2-2}$$

At the decoder side, the above process is reversed to compute $Q(r_{i,j}),\ 0 \le i \le M-1,\ 0 \le j \le N-1$, as follows:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j}, \text{ if vertical BDPCM is used} \tag{2-3}$$

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{i,k}, \text{ if horizontal BDPCM is used} \tag{2-4}$$

The inverse quantized residuals, $Q^{-1}\left(Q\left(r_{i,j}\right)\right)$, are added to the intra block prediction values to produce the reconstructed sample values.

The predicted quantized residual values $\tilde{r}_{i,j}$ are sent to the decoder using the same residual coding process as that in transform skip mode residual coding. For lossless coding, if slice_ts_residual_coding_disabled_flag is set to 1, the quantized residual values are sent to the decoder using regular transform residual coding as described in **2.2.2**. In terms of the MPM mode for future intra mode coding, horizontal or vertical prediction mode is stored for a BDPCM-coded CU if the BDPCM prediction direction is horizontal or vertical, respectively. For deblocking, if both blocks on the sides of a block boundary are coded using BDPCM, then that particular block boundary is not deblocked.

2.1.3 Residual coding for transform skip mode

VVC allows the transform skip mode to be used for luma blocks of size up to MaxTsSize by MaxTsSize, where the value of MaxTsSize is signaled in the PPS and can be at most 32. When a CU is coded in transform skip mode, its prediction residual is quantized and coded using the transform skip residual coding process. This process is modified from the transform coefficient coding process described in **2.2.2**. In transform skip mode, the residuals of a TU are also coded in units of non-overlapped subblocks of size 4x4. For better coding efficiency, some modifications are made to customize the residual coding process towards the residual signal's characteristics. The following summarizes the differences between transform skip residual coding and regular transform residual coding:

- Forward scanning order is applied to scan the subblocks within a transform block and also the positions within a subblock;
- no signalling of the last (x, y) position;
- coded_sub_block_flag is coded for every subblock except for the last subblock when all previous flags are equal to 0;
- sig_coeff_flag context modelling uses a reduced template, and context model of sig_coeff_flag depends on top and left neighbouring values;
- context model of abs_level_gt1 flag also depends on the left and top sig_coeff_flag values;
- par_level_flag using only one context model;
- additional greater than 3, 5, 7, 9 flags are signalled to indicate the coefficient level, one context for each flag;

- rice parameter derivation using fixed order = 1 for the binarization of the remainder values;
- context model of the sign flag is determined based on left and above neighbouring values and the sign flag is parsed after sig_coeff_flag to keep all context coded bins together.

For each subblock, if the coded_subblock_flag is equal to 1 (i.e., there is at least one non-zero quantized residual in the subblock), coding of the quantized residual levels is performed in three scan passes (see Fig. 5):

- **First scan pass**: significance flag (sig_coeff_flag), sign flag (coeff_sign_flag), absolute level greater than 1 flag (abs_level_gtx_flag[0]), and parity (par_level_flag) are coded. For a given scan position, if sig_coeff_flag is equal to 1, then coeff_sign_flag is coded, followed by the abs_level_gtx_flag[0] (which specifies whether the absolute level is greater than 1). If abs_level_gtx_flag[0] is equal to 1, then the par_level_flag is additionally coded to specify the parity of the absolute level.
- **Greater-than-x scan pass**: for each scan position whose absolute level is greater than 1, up to four abs_level_gtx_flag[i] for i = 1...4 are coded to indicate if the absolute level at the given position is greater than 3, 5, 7, or 9, respectively.
- **Remainder scan pass**: The remainder of the absolute level abs_remainder are coded in bypass mode. The remainder of the absolute levels are binarized using a fixed rice parameter value of 1.

The bins in scan passes #1 and #2 (the first scan pass and the greater-than-x scan pass) are context coded until the maximum number of context coded bins in the TU have been exhausted. The maximum number of context coded bins in a residual block is limited to 1.75*block_width*block_height, or equivalently, 1.75 context coded bins per sample position on average. The bins in the last scan pass (the remainder scan pass) are bypass coded. A variable, RemCcbs, is first set to the maximum number of context-coded bins for the block and is decreased by one each time a context-coded bin is coded. While RemCcbs is larger than or equal to four, syntax elements in the first coding pass, which includes the sig_coeff_flag, coeff_sign_flag, abs_level_gt1_flag and par_level_flag, are coded using context-coded bins. If RemCcbs becomes smaller than 4 while coding the first pass, the remaining coefficients that have yet to be coded in the first pass are coded in the remainder scan pass (pass #3).

After completion of first pass coding, if RemCcbs is larger than or equal to four, syntax elements in the second coding pass, which includes abs_level_gt3_flag, abs_level_gt5_flag, abs_level_gt7_flag, and abs_level_gt9_flag, are coded using context coded bins. If the RemCcbs becomes smaller than 4 while coding the second pass, the remaining coefficients that have yet to be coded in the second pass are coded in the remainder scan pass (pass #3).

Fig. 5 illustrates the transform skip residual coding process. The star marks the position when context coded bins are exhausted, at which point all remaining bins are coded using bypass coding.

Further, for a block not coded in the BDPCM mode, a level mapping mechanism is applied to transform skip residual coding until the maximum number of context coded bins has been reached. Level mapping uses the top and left neighbouring coefficient levels to predict the current coefficient level in order to reduce signalling cost. For a given residual position, denote *absCoeff* as the absolute coefficient level before mapping and *absCoeffMod* as the coefficient level after mapping. Let X_0 denote the absolute coefficient level of the left neighbouring position and let X_1 denote the absolute coefficient level of the above neighbouring position. The level mapping is performed as follows:

```
pred = max(X0, X1);
if (absCoeff = = pred)
    absCoeffMod = 1;
else
    absCoeffMod = (absCoeff < pred) ? absCoeff + 1 : absCoeff.
```

Then, the absCoeffMod value is coded as described above. After all context coded bins have been exhausted, level mapping is disabled for all remaining scan positions in the current block.

2.1.4 Palette mode

In VVC, the palette mode is used for screen content coding in all of the chroma formats supported in a 4:4:4 profile (that is, 4:4:4, 4:2:0, 4:2:2 and monochrome). When palette mode is enabled, a flag is transmitted at the CU level if the CU size is smaller than or equal to 64x64, and the amount of samples in the CU is greater than 16 to indicate whether palette mode is used. Considering that applying palette mode on small CUs introduces insignificant coding gain and brings extra complexity on the small blocks, palette mode is disabled for CU that are smaller than or equal to 16 samples. A palette coded coding unit (CU) is treated as a prediction mode other than intra prediction, inter prediction, and intra block copy (IBC) mode.

If the palette mode is utilized, the sample values in the CU are represented by a set of representative colour values. The set is referred to as the palette. For positions with sample values close to the palette colours, the palette indices are signalled. It is also possible to specify a sample that is outside the palette by signalling an escape symbol. For samples within the CU that are coded using the escape symbol, their component values are signalled directly using (possibly) quantized component values. This is illustrated in Fig. 6. The quantized escape symbol is binarized with fifth order Exp-Golomb binarization process (EG5).

For coding of the palette, a palette predictor is maintained. The palette predictor is initialized to 0 at the beginning of each slice for non-wavefront case. For WPP case, the palette predictor at the beginning of each CTU row is initialized to the predictor derived from the first CTU in the previous CTU row so that the initialization scheme between palette predictors and CABAC synchronization is unified. For each entry in the palette predictor, a reuse flag is signalled to indicate whether it is part of the current palette in the CU. The reuse flags are sent using run-length coding of zeros. After this, the number of new palette entries and the component values for the new palette entries are signalled. After encoding the palette coded CU, the palette predictor will be updated using the current palette, and entries from the previous palette predictor that are not reused in the current palette will be added at the end of the new palette predictor until the maximum size allowed is reached. An escape flag is signaled for each CU to indicate if escape symbols are present in the current CU. If escape symbols are present, the palette table is augmented by one and the last index is assigned to be the escape symbol.

In a similar way as the coefficient group (CG) used in transform coefficient coding, a CU coded with palette mode is divided into multiple line-based coefficient group, each consisting of *m* samples (i.e., m=16), where index runs, palette index values, and quantized colors for escape mode are encoded/parsed sequentially for each CG. Same as in HEVC, horizontal or vertical traverse scan can be applied to scan the samples, as shown in Fig. 7. Fig. 7 illustrates subblock-based index map scanning for palette, left for horizontal scanning and right for vertical scanning.

The encoding order for palette run coding in each segment is as follows: For each sample position, 1 context coded bin ***run_copy_flag = 0*** is signalled to indicate if the pixel is of the same mode as the previous sample position, i.e., if the previously scanned sample and the current sample are both of run type COPY_ABOVE or if the previously scanned sample and the current sample are both of run type INDEX and the same index value. Otherwise, ***run_copy_flag = 1*** is signaled. If the current sample and the previous sample are of different modes, one context coded bin ***copy_above_palette_indices_flag*** is signaled to indicate the run type, i.e., INDEX or COPY_ABOVE, of the current sample. Here, decoder doesn't have to parse run type if the sample is in the first row (horizontal traverse scan) or in the first column (vertical traverse scan) since the INDEX mode is used by default. With the same way, decoder doesn't have to parse run type if the previously parsed run type is COPY_ABOVE. After palette run coding of samples in one coding pass, the index values (for INDEX mode) and quantized escape colors are grouped and coded in another coding pass using CABAC bypass coding. Such separation of context coded bins and bypass coded bins can improve the throughput within each line CG.

For slices with dual luma/chroma tree, palette is applied on luma (Y component) and chroma (Cb and Cr components) separately, with the luma palette entries containing only Y values and the chroma palette entries containing both Cb and Cr values. For slices of single tree, palette will be applied on Y, Cb, Cr components jointly, i.e., each entry in the palette contains Y, Cb, Cr values, unless when a CU is coded using local dual tree, in which case coding of luma and chroma is handled separately. In this case, if the corresponding luma or choma blocks are coded using palette mode, their palette is applied in a way similar to the dual tree case (this is related to non-4:4:4 coding and will be further explained in 2.1.4.1).

For slices coded with dual tree, the maximum palette predictor size is 63, and the maximum palette table size for coding of the current CU is 31. For slices coded with dual tree, the maximum predictor and palette table sizes are halved, i.e., maximum predictor size is 31 and maximum table size is 15, for each of the luma palette and the chroma palette.For deblocking, the palette coded block on the sides of a block boundary is not deblocked.

2.1.4.1 Palette mode for non-4:4:4 content

Palette mode in VVC is supported for all chroma formats in a similar manner as the palette mode in HEVC SCC. For non-4:4:4 content, the following customization is applied:

1. When signaling the escape values for a given sample position, if that sample position has only the luma component but not the chroma component due to chroma subsampling, then only the luma escape value is signaled. This is the same as in HEVC SCC.
2. For a local dual tree block, the palette mode is applied to the block in the same way as the palette mode applied to a single tee block with two exceptions:
    a. The process of palette predictor update is slightly modified as follows. Since the local dual tree block only contains luma (or chroma) component, the predictor update process uses the signalled value of luma (or chroma) component and fills the "missing" chroma (or luma) component by setting it to a default value of (1 << (component bit depth - 1)).

   b. The maximum palette predictor size is kept at 63 (since the slice is coded using single tree) but the maximum palette table size for the luma/chroma block is kept at 15 (since the block is coded using separate palette).

3. For palette mode in monochrome format, the number of colour components in a palette coded block is set to 1 instead of 3.

2.1.4.2 Encoder algorithm for palette mode

At the encoder side, the following steps are used to produce the palette table of the current CU

1. First, to derive the initial entries in the palette table of the current CU, a simplified K-means clustering is applied. The palette table of the current CU is initialized as an empty table. For each sample position in the CU, the SAD between this sample and each palette table entry is calculated and the minimum SAD among all palette table entries is obtained. If the minimum SAD is smaller than a pre-defined error limit, errorLimit, then the current sample is clustered together with the palette table entry with the minimum SAD. Otherwise, a new palette table entry is created. The threshold errorLimit is QP-dependent and is retrieved from a look-up table containing 57 elements covering the entire QP range. After all samples of the current CU have been processed, the initial palette entries are sorted according to the number of samples clustered together with each palette entry, and any entry after the 31st entry is discarded.
2. In the second step, the initial palette table colours are adjusted by considering two options: using the centroid of each cluster from step 1 or using one of the palette colours in the palette predictor. The option with lower rate-distortion cost is selected to be the final colours of the palette table. If a cluster has only a single sample and the corresponding palette entry is not in the palette predictor, the corresponding sample is converted to an escape symbol in the next step.
3. A palette table thus generated contains some new entries from the centroids of the clusters in step 1, and some entries from the palette predictor. So this table is reordered again such that all new entries (i.e. the centroids) are put at the beginning of the table, followed by entries from the palette predictor.

Given the palette table of the current CU, the encoder selects the palette index of each sample position in the CU. For each sample position, the encoder checks the RD cost of all index values corresponding to the palette table entries, as well as the index representing the escape symbol, and selects the index with the smallest RD cost using the following equation.

$$\text{RD cost} = \text{distortion} \times (\text{isChroma? } 0.8 : 1) + \text{lambda} \times \text{bypass coded bits} \qquad (2\text{-}5)$$

After deciding the index map of the current CU, each entry in the palette table is checked to see if it is used by at least one sample position in the CU. Any unused palette entry will be removed. After the index map of the current CU is decided, trellis RD optimization is applied to find the best values of run_copy_flag and run type for each sample position by comparing the RD cost of three options: same as the previously scanned position, run type COPY_ABOVE, or run type INDEX. When calculating the SAD values, sample values are scaled down to 8 bits, unless the CU is coded in lossless mode, in which case the actual input bit depth is used to calculate the SAD. Further, in the case of lossless coding, only rate is used in the rate-distortion optimization steps mentioned above (because lossless coding incurs no distortion).

2.1.5 Adaptive color transform

In HEVC SCC extension, adaptive color transform (ACT) was applied to reduce the redundancy between three color components in 444 chroma format. The ACT is also adopted into the VVC standard to enhance the coding efficiency of 444 chroma format coding. Same as in HEVC SCC, the ACT performs in-loop color space conversion in the prediction residual domain by adaptively converting the residuals from the input color space to YCgCo space. Fig. 8 illustrates the decoding flowchart with the ACT being applied. Two color spaces are adaptively selected by signaling one ACT flag at CU level. When the flag is equal to one, the residuals of the CU are coded in the YCgCo space; otherwise, the residuals of the CU are coded in the original color space. Additionally, same as the HEVC ACT design, for inter and IBc CUs, the ACT is only enabled when there is at least one non-zero coefficient in the CU. For intra CUs, the ACT is only enabled when chroma components select the same intra prediction mode of luma component, i.e., DM mode.

2.1.5.1 ACT mode

In HEVC SCC extension, the ACT supports both lossless and lossy coding based on lossless flag (i.e.,cu_transquant_bypass_flag). However, there is no flag signalled in the bitstream to indicate whether lossy or lossless coding is applied. Therefore, YCgCo-R transform is applied as ACT to support both lossy and lossless cases. The YCgCo-R reversible colour transform is shown as below.

Forward Conversion: GBR to YCgCo	Backward Conversion: YCgCo to GBR
Co = R-B;	t = Y – (Cg>>1)

(2-6)

t = B + (Co>>1); Cg = G – t; Y = t + (Cg>>1);	G = Cg + t B = t – (Co>>1) R = Co + B

Since the YCgCo-R transform are not normalized. To compensate the dynamic range change of residuals signals before and after color transform, the QP adjustments of (−5, 1, 3) are applied to the transform residuals of Y, Cg and Co components, respectively. The adjusted quantization parameter only affects the quantization and inverse quantization of the residuals in the CU. For other coding processes (such as deblocking), original QP is still applied.

Additionally, because the forward and inverse color transforms need to access the residuals of all three components, the ACT mode is always disabled for separate-tree partition and ISP mode where the prediction block size of different color component is different. Transform skip (TS) and block differential pulse coded modulation (BDPCM), which are extended to code chroma residuals, are also enabled when the ACT is applied.

2.1.5.2 ACT fast encoding algorithms

To avoid brutal R-D search in both the original and converted color spaces, the following fast encoding algorithms are applied in the VTM reference software to reduce the encoder complexity when the ACT is enabled.

- The order of RD checking of enabling/disabling ACT is dependent on the original color space of input video. For RGB videos, the RD cost of ACT mode is checked first; for YCbCr videos, the RD cost of non-ACT mode is checked first. The RD cost of the second color space is checked only if there is at least one non-zero coefficient in the first color space.
- The same ACT enabling/disabling decision is reused when one CU is obtained through different partition path. Specifically, the selected color space for coding the residuals of one CU will be stored when the CU is coded at the first time. Then, when the same CU is obtained by another partition path, instead of checking the RD costs of the two spaces, the stored color space decision will be directly reused.
- The RD cost of a parent CU is used to decide whether to check the RD cost of the second color space for the current CU. For instance, if the RD cost of the first color space is smaller than that of the second color space for the parent CU, then for the current CU, the second color space is not checked.

- To reduce the number of tested coding modes, the selected coding mode is shared between two color spaces. Specifically, for intra mode, the preselected intra mode candidates based on SATD-based intra mode selection are shared between two color spaces. For inter and IBC modes, block vector search or motion estimation is performed only once. The block vectors and motion vectors are shared by two color spaces.

2.1.6 Intra template matching

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.

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. 9 consisting of:

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

SAD is used as a cost function.

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.

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.

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.

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

2.1.7 IBC with template matching (IBC-TM)

In ECM-5.0, Template Matching with IBC is used for both IBC merge mode and IBC AMVP mode.

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), where W is the width and H the height of the current CU.

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.

In the IBC-TM AMVP mode, up to 3 candidates are selected from the IBC-TM 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.

The Template Matching refinement for both IBC-TM merge and AMVP modes is quite simple since IBC motion vectors are constrained (i) to be integer and (ii) within a reference region as shown in Fig. 10. 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 depending on the AMVR value. Such a refinement accesses only to samples without interpolation. In both cases, the refined motion vectors and the used template in each refinement step must respect the constraint of the reference region.

2.1.8 Enlarged HMVP table for IBC

In ECM-5.0, the HMVP table size for IBC is increased to 25. After up to 20 IBC merge candidates are derived with full pruning, they are reordered together. After reordering, the first 6 candidates with the lowest template matching costs are selected as the final candidates in the IBC merge list.

2.1.9 Block Vector Difference Binarization

In ECM-4.0, Block Vector Difference (BVD) shares the same binarization method with Motion Vector Difference (MVD). For each component, greater than 0, greater than 1 flags are signalled followed by the bypass coded remaining magnitude binarized with EG1 code.

In ECM-5.0, the greater than 1 flag is removed and the first 5 bins of the EG1 prefix are context coded, all other bins are kept bypass coded.

2.1.10 Reconstruction-Reordered IBC (RRIBC)

At the JVET-Z meeting, a Reconstruction-Reordered IBC (RR-IBC) mode was 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.

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.

To better utilize the symmetry property, a flip-aware BV adjustment approach is applied to refine the block vector candidate. Fig. 11A illustrates an example of BV adjustment for a horizontal flip, and Fig. 11B illustrates an example of BV adjustment for a vertical flip. For example, as shown in Figs. 11A and 11B, (x_{nbr}, y_{nbr}) and (x_{cur}, y_{cur}) represent the coordinates of the center sample of the neighbouring block and the current block, respectively, BV^{nbr} and BV^{cur} denotes the BV of the neighbouring 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., $BV^{cur}{}_{v} = 2(y_{nbr} - y_{cur}) + BV^{nbr}{}_{v}$.

2.1.11 IBC Merge Mode with Block Vector Differences

In this disclosure, IBC merge mode with block vector differences (a.k.a. IBC-MBVD) is introduced. Similar as regular MMVD mode, in IBC-MBVD, after an IBC base candidate is selected, it is further refined by the signalled BVDs information.

In method #1, the distance set is {1-pel, 2-pel, 4-pel, 8-pel, 16-pel, 32-pel, 48-pel, 64-pel, 80-

pel, 96-pel, 112-pel, 128-pel}, and the BVD directions are two horizontal and two vertical directions.

In method #2, 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.

For both methods, the base candidates are selected from 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 (12×4 for method #1, 20×4 for method #2) for each base candidate are reordered. Finally, the top 1/4 refinement positions with the lowest template SAD costs are kept as available positions, consequently for MBVD index coding.

2.2 Sample reordering

2.2.1 Reordering of Reconstruction sample

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

In the following disclosure, a block may refer to a coding block (CB), a coding unit (CU), a prediction block (PB), a prediction unit (PU), a transform block (TB), a transform unit (TU), a sub-block, a sub-CU, a coding tree unit (CTU), a coding tree block (CTB), or a coding group (CG).

In the following disclosure, a region may refer to any video unit, such as a picture, a slice or a block. A region may also refer to a non-rectangular region, such as a triangular.

In the following disclosure, W and H represents the width and height of a mentioned rectangular region.

1. It is proposed that the samples in a region may be reordered.
    a. Reordering of samples may be defined as following: Suppose the sample at position (x, y) in a region before reordering is denoted as S(x, y), and the sample at position (x, y) in a region after reordering is denoted as R(x, y). It is required that R(x, y) = S(f(x, y), g(x, y)), wherein (f(x, y), g(x, y)) is a position in the region, f and g are two functions.
        i. For example, it is required that there is at least one position (x, y) satisfying that (f(x, y), g(x, y)) is not equal to (x, y).

- b. The samples in a region to be reordered may be:
    - i. Original samples before encoding;
    - ii. Prediction samples;
    - iii. Reconstruction samples;
    - iv. Transformed samples (transformed coefficients);
    - v. Samples before inverse-transform (coefficients before inverse-transform);
    - vi. Samples before deblocking filtering;
    - vii. Samples after deblocking filtering;
    - viii. Samples before SAO processing;
    - ix. Samples after SAO processing;
    - x. Samples before ALF processing;
    - xi. Samples after ALF processing;
    - xii. Samples before post processing;
    - xiii. Samples after post processing.
- c. In one example, reordering may be applied at more than one stage.
    - i. For example, at least two of these samples listed in bullet 1.b may be reordered.
        - 1) For example, different reordering methods may be applied on the two kinds of samples.
        - 2) For example, the same reordering method may be applied on the two kinds of samples.
- d. In one example, reordering may be a horizontal flip. For example, $f(x, y)= P-x$, $g(x, y) = y$. E.g. $P = W -1$.
- e. In one example, reordering may be a vertical flip. For example, $f(x, y)= x$, $g(x, y) = Q - y$. E.g. $Q = H -1$.
- f. In one example, reordering may be a horizontal-vertical flip. For example, $f(x, y)= P-x$, $g(x, y) = Q - y$. E.g. $P = W -1$ and $Q = H -1$.
- g. In one example, reordering may be a shift. For example, $f(x, y)= (P+x)\%W$, $g(x, y) = (Q + y) \% H$, wherein P and Q are integers.
- h. In one example, reordering may be a rotation.
- i. In one example, there is at least one (x, y) satisfying (x, y) is equal to (f(x,y), g(x,y)).
- j. In one example, whether to and/or how to reorder the samples may be signaled from the encoder to the decoder, such as in SPS/sequence header/PPS/picture header/APS/slice header/sub-picture/tile/CTU line/CTU/CU/PU/TU.

        i. For example, a first flag is signaled to indicate whether reordering is applied.
            1) E.g., the first flag may be coded with context coding.
        ii. For example, a second syntax element (such as a flag) is signaled to indicate which reordering method is used (such as horizontal flip or vertical flip).
            1) E.g. the second syntax element is signaled only if it is indicated that reordering is applied.
            2) E.g., the second syntax element may be coded with context coding.

2. It is proposed that whether to and/or how to reorder the samples may depend on coding information.
    a. In one example, whether to and/or how to reorder the samples may be derived depending on coding information at picture level/slice level/CTU level/CU level/PU level/TU level.
    b. In one example, the coding information may comprise:
        i. Dimensions of the region.
        ii. Coding mode of the region (such as inter, intra or IBC).
        iii. Motion information (such as motion vectors and reference indices).
        iv. Intra-prediction mode (such as angular intra-prediction mode, Planar or DC).
        v. Inter-prediction mode (such as affine prediction, bi-prediction/uni-prediction, merge mode, combined inter-intra prediction (CIIP), merge with motion vector difference (MMVD), temporal motion vector prediction (TMVP), sub-TMVP).
        vi. Quantization parameter (QP).
        vii. Coding tree splitting information such as coding tree depth.
        viii. Color format and/or color component.
3. It is proposed that at least one parsing or decoding procedure other than the reordering procedure may depend on whether to and/or how to reorder samples.
    a. For example, a syntax element may be signaled conditionally based on whether reordering is applied or not.
    b. For example, different scanning order may be used based on whether to and/or how to reorder samples.
    c. For example, deblocking filtering/SAO/ALF may be used based on whether to and/or how to reorder samples.

4. In one example, samples may be processed by at least one auxiliary procedure before or after the resampling process. Some possible auxiliary procedures may comprise: (combination may be allowed)
    a. For example, at least one sample may be added by an offset.
    b. For example, at least one sample may be multiplied by a factor.
    c. For example, at least one sample may be clipped.
    d. For example, at least one sample may be filtered.
    e. For example, at least one sample X may be modified to be T(X), wherein T is a function.
5. In one example, for a block coded with IBC mode.
    a. For example, a first flag is signaled to indicate whether reconstruction samples should be reordered.
        i. E.g., the first flag may be coded with context coding.
    b. For example, a second flag may be signaled to indicate whether reconstruction samples should be flipped horizontally or vertically.
        i. E.g. the second flag is signaled only if the first flag is true.
        ii. E.g., the second flag may be coded with context coding.

2.2.2 On sample reordering – application condition, and the interaction with other procedures

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

The terms ‘video unit’ or ‘coding unit’ may represent a picture, a slice, a tile, a coding tree block (CTB), a coding tree unit (CTU), a coding block (CB), a CU, a PU, a TU, a PB, a TB.

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.

It is noted that the terminologies mentioned below are not limited to the specific ones defined in existing standards. Any variance of the coding tool is also applicable.

1. About the application condition of sample reordering (e.g., the 1st and related issues), the following methods are proposed.
    a. Whether a reordering process is applied on a reconstruction/original/prediction block may be dependent on coded information of a video unit.
        a. For example, it may depend on the prediction method.

- b. For example, if a video unit is coded with to one or more modes/techniques as listed below, the reordering process may be applied to the video unit. Otherwise, reordering process is disallowed.
  - i. Intra block copy (a.k.a., IBC).
  - ii. Current picture referencing (a.k.a., CPR).
  - iii. Intra template matching (a.k.a., IntraTM).
  - iv. IBC template matching (or template matching based IBC mode).
  - v. Merge based coding.
  - vi. AMVP based coding.
- c. For example, it may depend on block dimensions (such as block width and/or height).
- d. For example, if the dimensions W×H of the video unit conform to one or more rules as listed below, the reordering process may be applied to the video unit. Otherwise, reordering process is disallowed.
  - i. if W >=T1 and/or H>=T2.
  - ii. if W <=T1 and/or H<=T2.
  - iii. if W >T1 and/or H>T2.
  - iv. if W <T1 and/or H<T2.
  - v. if W×H >=T.
  - vi. if W×H >T.
  - vii. if W×H <=T.
  - viii. if W×H <T.

2. About what kind of samples are reordered and the interaction with other procedures (e.g., the 2th and related issues), the following methods are proposed.
   - a. A possible sample reordering method may refer to one or more processes as followings:
     - a. Reshaper domain samples (e.g., obtained based on LMCS method) of a video unit may be reordered.
       - i. For example, reshaper domain luma samples (e.g., obtained based on luma mapping of the LMCS method) of a video unit may be reordered.
     - b. The original domain (rather than LMCS reshaper domain) samples of a video unit may be reordered.
       - i. For example, original domain chroma samples of a video unit may be reordered.

    ii. For example, original domain luma samples of a video unit may be reordered.

c. Reconstruction samples of a video unit may be reordered.
    i. For example, reconstruction samples of the video unit may be reordered right after adding decoded residues to predictions.
    ii. For example, reshaper domain luma reconstruction samples of the video unit may be reordered.
    iii. For example, original domain luma reconstruction samples of the video unit may be reordered.
    iv. For example, original domain chroma reconstruction samples of the video unit may be reordered.
d. Inverse luma mapping of LMCS process may be applied based on reordered reconstruction samples.
e. Loop filter process (e.g., luma/chroma bilateral filter, luma/chroma SAO, CCSAO, luma/chroma ALF, CCALF, etc.) may be applied based on reordered reconstruction samples.
    i. For example, loop filter process may be applied based on original domain (rather than LMCS reshaper domain) reordered reconstruction samples.
f. Distortion calculation (e.g., SSE computation between original samples and reconstruction samples) may be based on reordered reconstruction samples.
    i. For example, distortion calculation may be based on original domain reordered reconstruction samples.
g. Original samples of a video unit may be reordered.
    i. For example, the reshaper domain original luma samples of a video unit may be reordered.
    ii. For example, the original domain original luma samples of a video unit may be reordered.
    iii. For example, the original domain original chroma samples of a video unit may be reordered.
    iv. For example, the residues may be generated by subtracting the prediction from reordered original samples.
h. Prediction samples of a video unit may be reordered.

        i. For example, the reordering process for prediction samples may be performed right after the motion compensation process.
        ii. For example, sign prediction may be applied based on the reordered prediction samples of the video unit.

<u>General aspects</u>

3. 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.
4. Whether to and/or how to apply the disclosed methods above may be signalled at PB/TB/CB/PU/TU/CU/VPDU/CTU/CTU row/slice/tile/sub-picture/other kinds of region contain more than one sample or pixel.
5. Whether to and/or how to apply the disclosed methods above may be dependent on coded information, such as block size, colour format, single/dual tree partitioning, colour component, slice/picture type.

2.2.3 On sample reordering – sample reordering, signalling and storage

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

The terms ‘video unit’ or ‘coding unit’ may represent a picture, a slice, a tile, a coding tree block (CTB), a coding tree unit (CTU), a coding block (CB), a CU, a PU, a TU, a PB, a TB.

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.

It is noted that the terminologies mentioned below are not limited to the specific ones defined in existing standards. Any variance of the coding tool is also applicable.

1. About the signalling of sample reordering (e.g., the 1st and related issues), the following methods are proposed.
    a. For example, at least one new syntax elements (e.g., flag, index, variable, parameter, etc) may be signalled to specify the usage of sample reordering for a video unit.
        a. For example, at least one new syntax elements (e.g., a flag) may be further signalled to specify the usage of sample reordering, given that a certain prediction method is used to a video unit.
        b. For example, a first new syntax element (e.g., a flag) may be further signalled, specifying the usage of sample reordering for an intra template

    matching coded video unit, given that the intra template matching usage flag specifies the video unit is coded by intra template matching.

    c. For example, a first new syntax element (e.g., a flag) may be further signalled, specifying the usage of sample reordering for an IBC amvp coded video unit, given that the IBC amvp flag specifies the video unit is coded by IBC amvp.
    d. For example, a first new syntax element (e.g., a flag) may be further signalled, specifying the usage of sample reordering for an IBC merge coded video unit, given that the IBC merge flag specifies the video unit is coded by IBC merge.

b. Furthermore, for example, if the first new syntax element specifies the sample reordering is used to the certain prediction method coded video unit, then a second new syntax element (e.g., a flag) may be further signalled, specifying which reordering method (such as horizontal flipping or vertical flipping) is used to the video unit.
c. For example, a single new syntax element (e.g., a parameter, or a variable, or an index) may be signalled to a video unit, instead of multiple cascaded syntax elements, specifying the type of reordering (such as no flipping, horizontal flipping, or vertical flipping) applied to the video unit.
    a. For example, one new syntax element (e.g., an index) may be further signalled, specifying the type of sample reordering for an intra template matching coded video unit, given that the intra template matching usage flag specifies the video unit is coded by intra template matching.
    b. For example, one new syntax element (e.g., an index) may be further signalled, specifying the type of sample reordering for an IBC amvp coded video unit, given that the IBC amvp flag specifies the video unit is coded by IBC amvp.
    c. For example, one new syntax element (e.g., an index) may be further signalled, specifying the type of sample reordering for an IBC merge coded video unit, given that the IBC merge flag specifies the video unit is coded by IBC merge.
    d. Additionally, for example, the new syntax element (e.g., an index) equal to 0 specifies that no sample reordering is used; equal to 1 specifies that sample

reordering method A is used; equal to 2 specifies that sample reordering method B is used; and etc.

- d. For example, one or more syntax elements related to sample reordering may be context coded.
  - a. For example, the context may be based on neighboring blocks/samples coding information (e.g., such as availability, prediction mode, where or not merge coded, whether or not IBC coded, whether or not apply sample reordering, which sample reordering method is used, and etc.).
- e. Alternatively, for example, instead of signalling whether to do the sample reordering and/or which reordering method is used to a video unit, partial (or all) of these steps may be determined based on pre-defined rules (without signalling).
  - a. For example, the pre-defined rules may be based on neighboring blocks/samples coded information.
  - b. For example, given that the IBC merge flag specifies the video unit is coded by IBC merge, a procedure may be conducted to determine whether to perform reordering and how to reorder, based on pre-defined rules/procedures without signalling.
    - i. Alternatively, for example, given that the first new syntax element specifies the sample reordering is used to the video unit, however, instead of further signalling the reordering method, how to reorder may be determined based on pre-defined rules/procedures (without signalling).
    - ii. Alternatively, for example, whether to perform reordering may be implicit determined based on pre-defined rules/procedures, but how to reorder may be signalled.
  - c. For example, given that the IBC amvp flag specifies the video unit is coded by IBC amvp, a procedure may be conducted to determine whether to perform reordering and how to reorder, based on pre-defined rules/procedures without signalling.
    - i. Alternatively, for example, given that the first new syntax element specifies the sample reordering is used to the video unit, however, instead of further signalling the reordering method, how to reorder may be determined based on pre-defined rules/procedures (without signalling).

            - ii. Alternatively, for example, whether to perform reordering may be implicit determined based on pre-defined rules/procedures, but how to reorder may be signalled.
        - d. For example, given that the intra template matching flag specifies the video unit is coded by IBC merge, a procedure may be conducted to determine whether to perform reordering and how to reorder, based on pre-defined rules/procedures without signalling.
            - i. Alternatively, for example, given that the first new syntax element specifies the sample reordering is used to the video unit, however, instead of further signalling the reordering method, how to reorder may be determined based on pre-defined rules/procedures (without signalling).
            - ii. Alternatively, for example, whether to perform reordering may be implicit determined based on pre-defined rules/procedures, but how to reorder may be signalled.
    - f. For example, whether to perform reordering and/or how to reorder may be inherited from coded blocks.
        - a. For example, it may be inherited from an adjacent spatial neighbor block.
        - b. For example, it may be inherited from a non-adjacent spatial neighbor block.
        - c. For example, it may be inherited from a history-based motion table (such as a certain HMVP table).
        - d. For example, it may be inherited from a temporal motion candidate.
        - e. For example, it may be inherited based on an IBC merge candidate list.
        - f. For example, it may be inherited based on an IBC amvp candidate list.
        - g. For example, it may be inherited based on a generated motion candidate list/table.
        - h. For example, the sample reordering inheritance may be allowed in case that a video unit is coded by IBC merge mode.
        - i. For example, the sample reordering inheritance may be allowed in case that a video unit is coded by IBC AMVP mode.
        - j. For example, the sample reordering inheritance may be allowed in case that a video unit is coded by intra template matching mode.

2. About the storage of sample reordering status (e.g., the 2nd and related issues), the following methods are proposed:

   a. For example, the information of whether and/or how to reorder for a video unit may be stored.
      a. For example, the stored information may be used for future video unit's coding.
      b. For example, the information may be stored in a buffer.
         i. For example, the buffer may be a line buffer, a table, more than one line buffer, picture buffer, compressed picture buffer, temporal buffer, etc.
      c. For example, the information may be stored in a history motion vector table (such as a certain HMVP table).
   b. For example, coding information (e.g., such as whether or not apply sample reordering, which sample reordering method is used, block availability, prediction mode, where or not merge coded, whether or not IBC coded, and etc.) may be stored for the derivation of the context of sample reordering syntax element(s).

<u>General aspects</u>

3. 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.
4. Whether to and/or how to apply the disclosed methods above may be signalled at PB/TB/CB/PU/TU/CU/VPDU/CTU/CTU row/slice/tile/sub-picture/other kinds of region contain more than one sample or pixel.
5. Whether to and/or how to apply the disclosed methods above may be dependent on coded information, such as block size, colour format, single/dual tree partitioning, colour component, slice/picture type.

2.2.4 On sample reordering - motion list generation, implicit derivation, and how to reorder

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

The terms 'video unit' or 'coding unit' may represent a picture, a slice, a tile, a coding tree block (CTB), a coding tree unit (CTU), a coding block (CB), a CU, a PU, a TU, a PB, a TB.

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.

It is noted that the terminologies mentioned below are not limited to the specific ones defined in existing standards. Any variance of the coding tool is also applicable.

1. About the motion candidate list generation for sample reordering (e.g., the 1st and related issues), the following methods are proposed.
    - b. For example, IBC merge motion candidate list may be used for both regular IBC merge mode and sample reordering based IBC merge mode.
    - c. For example, IBC amvp motion predictor candidate list may be used for both regular IBC amvp mode and sample reordering based IBC amvp mode.
    - d. For example, a new motion (predictor) candidate list may be generated for a target video unit coded with sample reordering.
        - a. For example, the new candidate list may only consider motion candidates with same reordering method as the reordering method of the target video unit.
        - b. For example, the new candidate list may only consider motion candidates coded with sample reordering (but no matter the type of sample reordering method).
        - c. Alternatively, the new candidate list may be generated without considering the sample reordering method of each motion candidate.
        - d. For example, non-adjacent motion candidates may be inserted to the new candidate list.
            - i. For example, non-adjacent candidates with sample reordering (but no matter the type of sample reordering method) may be inserted.
            - ii. For example, non-adjacent candidates with same reordering method as the reordering method of the target video unit may be inserted.
            - iii. For example, non-adjacent candidates may be inserted no matter the sample reordering method is used to the candidate or not.
        - e. For example, new motion candidates may be generated according to a certain rule and inserted to the new candidate list.
            - i. For example, the rule may be based on averaging process.
            - ii. For example, the rule may be based on clipping process.
            - iii. For example, the rule may be based on scaling process.
    - e. For example, the motion (predictor) candidate list generation for a target video unit may be dependent on the reordering method.

    a. For example, the reordering method associated with each motion candidate (from spatial or temporal or history tables) may be inserted to the list, no matter the target video unit is to be coded with sample reordering or not.
    b. For example, if the target video unit is to be coded with sample reordering, only those motion candidates (from spatial or temporal or history tables) who coded with same reordering method as the reordering method of the target video unit are inserted to the list.
    c. For example, if the target video unit is to be coded with sample reordering, only those motion candidates (from spatial or temporal or history tables) who coded with sample reordering (but no matter the type of sample reordering method) are inserted to the list.
    d. For example, if the target video unit is to be coded WITHOUT sample reordering, those motion candidates (from spatial or temporal or history tables) who coded with same reordering method may not be inserted to the list.
    e. Alternatively, the motion list generation for a video unit may not be dependent on the reordering method associated with each motion candidate.

f. For example, the Adaptive Reordering of Merge Candidates (ARMC) of a video unit may be dependent on the reordering method.
    a. For example, if the target video unit is to be coded with sample reordering, the motion candidates who coded with same reordering method as the reordering method of the target video unit may be put prior to those motion candidates who coded with different reordering method.
    b. For example, if the target video unit is to be coded with sample reordering, the motion candidates who coded with sample reordering (but no matter the type of sample reordering method) may be put prior to those motion candidates who coded with different reordering method.
    c. For example, if the target video unit is to be coded WITHOUT sample reordering, the motion candidates who coded without reordering method may be put prior to those motion candidates who coded with reordering method.
    d. Alternatively, the ARMC may be applied to the video unit, no matter the reordering method associated with each motion candidate.

2. About the implicit determination of sample reordering (e.g., the 2nd and related issues), the following methods are proposed.
    a. Whether or not reordering the reconstruction/original/prediction samples of a video unit may be implicitly derived from coded information at both encoder and decoder.
        a. The implicit derivation may be based on costs/errors/differences calculated from coded information.
            i. For example, costs/errors/differences may be calculated based on template matching.
            ii. For example, the template matching may be conducted by comparing samples in a first template and a second template.
                1. For example, the first template is constructed by a group of pre-defined samples neighboring to current video unit, while the second template is constructed by a group of corresponding samples neighboring to a reference video unit.
                2. For example, the cost/error may refer to the accumulated sum of differences between samples in the first template and corresponding samples in the second template.
                    a. For example, the difference may be based on luma sample value.
                3. For example, the sample may refer to reconstruction sample, or a variant based on reconstruction sample.
                4. For example, the sample may refer to prediction sample, or a variant based on prediction sample.
        b. For example, a first cost may be calculated without reordering (denoted by Cost0), a second cost may be calculated with reordering (denoted by Cost1). Eventually, the minimum cost value among {Cost0, Cost1} is identified and the corresponding coding method (without reorder, or, reorder) is determined as the final coding method of the video unit.
        c. Alternatively, whether reordering the reconstruction/original/prediction samples of a video unit may be signalled in the bitstream.
            i. For example, it may be signalled by a syntax element (e.g., flag).
    b. Which reordering method is used to reorder the reconstruction/original/prediction samples may be implicitly derived from coded information at both encoder and decoder.

  - a. For example, whether horizontal flipping or vertical flipping.
  - b. The implicit derivation may be based on costs/errors/differences calculated from coded information.
    - i. For example, costs/errors/differences may be calculated based on template matching.
    - ii. For example, the template matching may be conducted by comparing samples in a first template and a second template.
      1. For example, the first template is constructed by a group of pre-defined samples neighboring to current video unit, while the second template is constructed by a group of corresponding samples neighboring to a reference video unit.
      2. For example, the cost/error may refer to the accumulated sum of differences between samples in the first template and corresponding samples in the second template.
         - a. For example, the difference may be based on luma sample value.
      3. For example, the sample may refer to reconstruction sample, or a variant based on reconstruction sample.
      4. For example, the sample may refer to prediction sample, or a variant based on prediction sample.
    - iii. For example, a first cost may be calculated without reordering method A (denoted by Cost0), a second cost may be calculated with reordering method B (denoted by Cost1). Eventually, the minimum cost value among {Cost0, Cost1} is identified and the corresponding coding method (reorder method A, reorder method B) is determined as the final coding method of the video unit.
  - c. Alternatively, which reordering method is used to reorder the reconstruction/original/prediction samples of a video unit may be signalled in the bitstream.
    - i. For example, it may be signalled by a syntax element (e.g., flag, or an index, or a parameter, or a variable).
- c. Whether or not AND which reordering method is used to reorder the reconstruction/original/prediction samples of a video unit may be implicitly derived from coded information at both encoder and decoder.

    a. For example, a first cost may be calculated without reordering (denoted by Cost0), a second cost may be calculated with reordering method A (denoted by Cost1); a third cost may be calculated with reordering method B (denoted by Cost2). Eventually, the minimum cost value among {Cost0, Cost1, Cost2} is identified and the corresponding coding method (without reorder, reorder method A, reorder method B) is determined as the final coding method of the video unit.

3. About how to reorder samples (e.g., the 3rd and related issues), the following methods are proposed.
   b. A possible sample reordering method may refer to one or more processes as followings:
      a. The reordering process may be applied based on video units.
         i. For example, the reordering process may be based on a block/CU/PU/TU.
         ii. For example, the reordering process may not be based on a tile/slice/picture.
      b. Samples of a video unit may be transformed according to a M-parameter model (such as M = 2 or 4 or 6 or 8).
      c. Samples of a video unit may be reordered.
      d. Samples of a video unit may be rotated.
      e. Samples of a video unit may be transformed according to an affine model.
      f. Samples of a video unit may be transformed according to a linear model.
      g. Samples of a video unit may be transformed according to a projection model.
      h. Samples of a video unit may be flipped along the horizontal direction.
      i. Samples of a video unit may be flipped along the vertical direction.

<u>General aspects</u>

4. 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.
5. Whether to and/or how to apply the disclosed methods above may be signalled at PB/TB/CB/PU/TU/CU/VPDU/CTU/CTU row/slice/tile/sub-picture/other kinds of region contain more than one sample or pixel.

6. Whether to and/or how to apply the disclosed methods above may be dependent on coded information, such as block size, colour format, single/dual tree partitioning, colour component, slice/picture type.

3 Problems

How to handle the interaction between RRIBC and template matching based IBC (e.g., IBC-TM) needs to be considered.

1. In ECM-5.0, IBC-TM-MERGE mode is included and there is no RRIBC in the ECM code, so it is unknown how to build an IBC-TM-MERGE list for IBC-TM-MERGE mode in case that a neighbor candidate is coded as RRIBC.
2. In ECM-5.0, due to the usage of IBC-TM to the IBC-AMVP mode, TM is mandatorily used to refine every IBC-AMVP candidate, and there is no RRIBC in the ECM code. Therefore, how to build a RRIBC based IBC-AMVP list when IBC-TM is allowed is undefined.

How to handle the interaction between RRIBC and IBC with merge-based motion vector difference (e.g., IBC_MBVD) needs to be considered.

1. If IBC_MBVD is allowed/used for screen content coding, how to interact with RRIBC such as whether derive IBC_MBVD motion and/or flip type from a RRIBC coded neighbor block may be defined.
2. If IBC_MBVD is allowed/used for screen content coding, how to interact with RRIBC such as the design of MBVD offsets regarding the RRIBC flip type may be defined.

4 Detailed Solutions

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

The terms ‘video unit’ or ‘coding unit’ may represent a picture, a slice, a tile, a coding tree block (CTB), a coding tree unit (CTU), a coding block (CB), a CU, a PU, a TU, a PB, a TB.

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.

It is noted that the terminologies mentioned below are not limited to the specific ones defined in existing standards. Any variance of the coding tool is also applicable.

4.1 About the interaction between RRIBC and IBC-TM such as how to derive the motion data and flip type for an IBC-TM-MERGE coded block (e.g., the 1st problem and related issues), the following methods are proposed.

- a. For example, an IBC-TM-MERGE candidate may inherit motion from a RRIBC coded neighbor block.
    - a. In one example, the motion may be directly inherited.
    - b. In one example, the motion may be firstly adjusted (e.g., adding a motion shift to it) then inherited.
    - c. Alternatively, an IBC-TM-MERGE candidate may NOT inherit motion from a RRIBC coded neighbor block.
- b. For example, an IBC-TM-MERGE candidate may NOT inherit the flip type from a RRIBC coded neighbor block.
    - a. In one example, the flip type of the IBC-TM-MERGE candidate may always set equal to NO_FLIP, no matter whether or not the motion of such IBC-TM merge candidate is inherited from a RRIBC coded neighbor block.
- c. Alternatively, an IBC-TM-MERGE candidate may inherit flip type from a RRIBC coded neighbor block.
    - a. In one example, the flip type of such IBC-TM-MERGE candidate is set equal to the flip type of the RRIBC coded neighbor block.
- d. For example, an IBC-TM-MERGE candidate may inherit motion from a RRIBC coded neighbor block, but never inherit the flip type from the RRIBC coded neighbor block.
    - a. In one example, the motion may be directly inherited.
    - b. In one example, the motion may be firstly adjusted (e.g., adding a motion shift to it) then inherited.
    - c. In one example, the flip type of an IBC-TM-MERGE candidate may always set equal to NO_FLIP.
- e. For example, an IBC-TM-MERGE candidate may be prohibited to be derived based on a RRIBC coded neighbor block.
    - a. In such case, the motion and flip type of a RRIBC coded neighbor block may be prohibited to be added to an IBC-TM-MERGE candidate.
    - b. In such case, never insert a neighbor block's coding information to an IBC-TM-MERGE list, if the neighbor block is coded with RRIBC.

f. For example, an IBC-TM-MERGE candidate/block may be RRIBC coded.
    a. In one example, the motion data (motion vectors, reference indexes, and/or flip type) of a RRIBC coded IBC-TM-MERGE block may be inherited from a RRIBC coded neighbor block.
    b. For example, additionally, the motion vector inherited from a neighbor block may be adjusted by a motion adjustment process.
        i. For example, additionally, the adjusted motion vector may be further refined by template matching based methods.
    c. For example, additionally, the motion vector inherited from a neighbor block may be further refined by template matching based methods.
        i. For example, additionally, the refined motion vector may be further adjusted by a motion adjustment process.
    d. For example, the motion adjustment process may be based on the flip type and/or the positions/locations/coordination of current and neighbor block.
        i. For example, the motion adjustment may along one direction (either horizontal or vertical).
        ii. For example, horizontal adjustment may be applied for horizontal flip.
        iii. For example, vertical adjustment may be applied for vertical flip.
    e. For example, the motion refinement (e.g., TM based) offset of a RRIBC coded IBC-TM-MERGE block/candidate may be dependent on flip type.
        i. In one example, only horizontal motion vector offsets are allowed for a horizontally flipped RRIBC coded IBC-TM-MERGE block/candidate (e.g., FLIP_TYPE = FLIP_HOR).
            1. Moreover, alternatively, the vertical component of a motion vector offset may be required to be equal to 0, in case that the IBC-TM-MERGE block is coded with horizontal flip RRIBC.
        ii. In one example, only vertical motion vector offsets are allowed for a vertically flipped RRIBC coded IBC-TM-MERGE block/candidate (e.g., FLIP_TYPE = FLIP_VER).
            1. Moreover, alternatively, the horizontal component of a motion vector offset may be required to be equal to 0, in

case that the IBC-TM-MERGE block/candidate is coded with vertical flip RRIBC.

4.2 About the interaction between RRIBC and IBC-TM such as how to build a RRIBC based IBC-AMVP list when IBC-TM is allowed (e.g., the 2nd problem and related issues), the following methods are proposed.

a. For example, in case that the current video unit is IBC-AMVP mode with RRIBC (e.g., the flip type of the current video unit is NOT equal to NO_FLIP), an AMVP candidate may NOT be allowed to be refined by template matching (e.g., IBC-TM-AMVP).
b. For example, in case that the current video unit is IBC-AMVP mode with non-RRIBC (e.g., the flip type of the current video unit is equal to NO_FLIP), an AMVP candidate may be refined by template matching (e.g., IBC-TM AMVP).
c. For example, the MVD thresholds for the similarity check (e.g., pruning process by comparing the similarity between a potential candidate and another candidate already in the list) during IBC-AMVP list generation, may be different from different video units, according to whether the current video unit is coded by RRIBC, or non-RRIBC.
    a. In one example, assume the MVD threshold for an IBC-AMVP non-RRIBC coded video unit is equal to K1, and the MVD threshold for an IBC-AMVP RRIBC video unit is equal to K2, then, K1 may not be equal to K2.
    b. Additionally, furthermore, K1 may be greater than K2.
    c. Additionally, furthermore, K1 and/or K2 may be pre-defined.
        i. For example, K1 and/or K2 may be equal to a certain number (such as 0 or 1).
        ii. For example, K1 and/or K2 may be dependent on dimensions of the current video unit (such as width/height, number of samples/pixels).
        iii. For example, K1 and/or K2 may be derived based on the same rule used in the similarity check of an existing coding tool in the codec (e.g., IBC-TM-MERGE mode, inter-TM mode, and etc.).
    d. Alternatively, K1 may be equal to K2.

d. For example, in case that the current video unit is IBC-AMVP mode with RRIBC (e.g., the flip type of the current video unit is NOT equal to NO_FLIP) and an MVP candidate of the current video unit is also coded with RRIBC.
    a. In one example, the motion vector of the MVP candidate may be adjusted first then used for the current video unit.
    b. In one example, only if the flip type of the current video unit and the flip type of the neighbor block which used to derive the MVP candidate are same, the motion adjustment may be performed.
        i. Alternatively, as long as the current video unit and the neighbor block are RRIBC coded, the motion adjustment may be performed.
    c. In one example, the motion adjustment may refer to add a motion shift to the MVP candidate.
    d. In one example, the motion shift may be dependent on the block dimensions and/or locations of the current video unit.
    e. In one example, the motion shift may be dependent on the block dimensions and/or locations of the neighbor block which used to derive the MVP candidate.

e. For example, in case that the current video unit is IBC-AMVP mode with non-RRIBC (e.g., the flip type of the current video unit is equal to NO_FLIP) and an MVP candidate of the current video unit is coded with RRIBC.
    a. In one example, the motion vector of the MVP candidate may NOT be adjusted.
    b. Alternatively, the motion vector of the MVP candidate may be adjusted (e.g., by a motion adjustment process).
        i. In one example, as long as the neighbor block is RRIBC coded, the motion adjustment may be used.
        ii. For example, additionally, the adjusted motion vector may be further refined by template matching based methods.
    c. In one example, the motion adjustment may refer to add a motion shift to the MVP candidate.
    d. In one example, the motion shift may be dependent on the block dimensions and/or locations of the current video unit.

e. In one example, the motion shift may be dependent on the block dimensions and/or locations of the neighbor block which used to derive the MVP candidate.

f. In one example, additionally, the motion vector candidate inherited from a neighbor block may be further refined by template matching based methods.
    i. For example, additionally, the refined motion vector may be further adjusted by a motion adjustment process.

g. In one example, the motion adjustment process may be based on the flip type and/or the positions/locations/coordination of current and neighbor block.
    i. For example, the motion adjustment may along one direction (either horizontal or vertical).
    ii. For example, horizontal adjustment may be applied for horizontal flip.
    iii. For example, vertical adjustment may be applied for vertical flip.

h. In one example, the motion refinement (e.g., TM based) offset of a RRIBC coded IBC-TM-AMVP block/candidate may be dependent on flip type.
    i. In one example, only horizontal motion vector offsets are allowed for a horizontally flipped RRIBC coded IBC-TM-AMVP block/candidate (e.g., FLIP_TYPE = FLIP_HOR).
        1. Moreover, alternatively, the vertical component of a motion vector offset may be required to be equal to 0, in case that the IBC-TM- AMVP block/candidate is coded with horizontal flip RRIBC.
    ii. In one example, only vertical motion vector offsets are allowed for a vertically flipped RRIBC coded IBC-TM-AMVP block/candidate (e.g., FLIP_TYPE = FLIP_VER).
        1. Moreover, alternatively, the horizontal component of a motion vector offset may be required to be equal to 0, in case that the IBC-TM-MERGE block/candidate is coded with vertical flip AMVP.

4.3 About the interaction between RRIBC and IBC_MBVD such as whether or not derive IBC_MBVD motion and/or flip type from RRIBC coded neighbor blocks for an IBC_MBVD coded block (e.g., the 3rd problem and related issues), the following methods are proposed.

a. For example, an IBC_MBVD coded block may derive motion from a RRIBC coded neighbor block.
    a. In one example, the motion of the RRIBC coded neighbor block may be directly inherited as a base motion for the IBC_MBVD block.
    b. In one example, the motion of the RRIBC coded neighbor block may be firstly adjusted by a motion adjustment process (e.g., adding a motion shift to it, based on the flip type), then the adjusted motion may be used as a base motion for the IBC_MBVD block.
    c. In one example, the motion of the RRIBC coded neighbor block may be directly inherited as a base motion for the IBC_MBVD block, and the final block vector after adding a block vector difference may be adjusted by a motion adjustment process (e.g., adding a motion shift to it, based on the flip type).
b. In one example, the motion adjustment process may be based on the flip type and/or the positions/locations/coordination of current and neighbor block.
    a. For example, the motion adjustment may along one direction (either horizontal or vertical).
    b. For example, horizontal adjustment may be applied for horizontal flip.
    c. For example, vertical adjustment may be applied for vertical flip.
c. For example, an IBC_MBVD coded block may NOT derive motion from a RRIBC coded neighbor block.
    a. In one example, the motion of a RRIBC coded neighbor block may not be used as a base motion for the IBC_MBVD block.
    b. For example, an IBC_MBVD coded block may be prohibited to be derived based on a RRIBC coded neighbor block.
        i. In such case, the motion and flip type of a RRIBC coded neighbor block may be prohibited to be inherited for the IBC_MBVD coded block.

        ii. In such case, never insert a neighbor block's coding information to IBC_MBVD-MERGE list, if the neighbor block is coded with RRIBC.

d. For example, an IBC_MBVD coded block may inherit the flip type from a RRIBC coded neighbor block.
    a. In one example, the flip type of such IBC_MBVD candidate is set equal to the flip type of the RRIBC coded neighbor block.
e. For example, an IBC_MBVD coded block may NOT inherit flip type from a RRIBC coded neighbor block.
    a. In one example, the flip type of the IBC_MBVD candidate may always set equal to NO_FLIP, no matter whether or not the base motion of such IBC_MBVD candidate is inherited from a RRIBC coded neighbor block.
f. For example, an IBC_MBVD coded block may derive both motion and flip type from a RRIBC coded neighbor block.
    a. Alternatively, an IBC_MBVD coded block may derive motion (but not flip type) from a RRIBC coded neighbor block.
    b. Alternatively, an IBC_MBVD coded block may inherit flip type (but not motion) from a RRIBC coded neighbor block.

4.4 About the interaction between RRIBC and IBC_MBVD such as how to design the MBVD offsets for an IBC_MBVD coded block (e.g., the 4th problem and related issues), the following methods are proposed.

a. For example, an MBVD offset may be added to a base motion candidate, and the allowable MBVD offsets associated to a RRIBC coded block (or base motion candidate) may be different from the allowable MBVD offsets associated to a non-RRIBC coded block (or base motion candidate).
    a) In one example, the allowable MBVD offsets for a RRIBC coded block may be a subset of those for a non-RRIBC coded block.
    b) In one example, the allowable MBVD offsets associated to a RRIBC coded base motion candidate may be a subset of those associated to a non-RRIBC coded base motion candidate.

   c) Alternatively, the allowable MBVD offsets associated to a RRIBC coded base motion candidate may be same as those associated to a non-RRIBC coded base motion candidate.

b. For example, what MBVD offsets are allowed for a coding block (or base motion candidate) may be dependent on the RRIBC flip type of such block (or base motion candidate).
   a) In one example, only horizontal MBVD offsets are allowed for a horizontally flipped RRIBC coded block (e.g., FLIP_TYPE = FLIP_HOR).
      i. Moreover, alternatively, the vertical component of MBVD offset may be required to be equal to 0, in case that the block is coded with horizontal flip RRIBC.
   b) In one example, additionally, only vertical MBVD offsets are allowed for a vertically flipped RRIBC coded block (e.g., FLIP_TYPE = FLIP_VER).
      i. Moreover, alternatively, the horizontal component of MBVD offset may be required to be equal to 0, in case that the block is coded with vertical flip RRIBC.
   c) In one example, only horizontal MBVD offsets are allowed for a horizontally flipped RRIBC base motion candidate (e.g., FLIP_TYPE = FLIP_HOR).
      i. Moreover, alternatively, the vertical component of MBVD offset may be required to be equal to 0, in case that the base motion candidate is coded with horizontal flip RRIBC.
   d) In one example, additionally, only vertical MBVD offsets are allowed for a vertically flipped RRIBC base motion candidate (e.g., FLIP_TYPE = FLIP_VER).
      i. Moreover, alternatively, the horizontal component of MBVD offset may be required to be equal to 0, in case that the base motion candidate is coded with vertical flip RRIBC.

c. For example, RRIBC based final IBC_MBVD motion vector candidates and non-RRIBC based final IBC_MBVD motion vector candidates may be grouped together.
   a) Furthermore, alternatively, an MMVD candidate index may be coded based on the group.

    - b) Furthermore, alternatively, a decoder side reordering (such as sorting) process may be applied to sort all candidates in the group, based on a cost (such as template cost).
  - d. For example, RRIBC based final IBC_MBVD motion vector candidates may be grouped together as a first subgroup, and non-RRIBC based final IBC_MBVD motion vector candidates may be grouped together as a second subgroup.
    - a) Furthermore, alternatively, an MMVD candidate index may be coded based on a specified subgroup.
    - b) Furthermore, alternatively, a decoder side reordering (such as sorting) process may be applied to sort candidates in each subgroup, respectively, based on a cost (such as template cost).
  - e. In one example, "RRIBC coded" means the flip type is NOT equal to NO_FLIP.
  - f. In one example, "non-RRIBC coded" means the flip type is equal to NO_FLIP.

General aspects

4.5 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.

4.6 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.

[0080] More details of the embodiments of the present disclosure will be described below which are related to the interaction of RRIBC and IBC template matching. The embodiments of the present disclosure should be considered as examples to explain the general concepts and should not be interpreted in a narrow way. Furthermore, these embodiments can be applied individually or combined in any manner.

[0081] As used herein, the term "block" may represent a color component, a sub-picture, a slice, a tile, a coding tree unit (CTU), a CTU row, groups of CTU, a coding unit (CU), a prediction unit (PU), a transform unit (TU), a coding tree block (CTB), a coding block (CB), a prediction block (PB), a transform block (TB), a sub-block of a video block, a sub-region within a video block, a video processing unit comprising multiple samples/pixels, and/or the like. A block may be rectangular or non-rectangular.

[0082] Fig. 12 illustrates a flowchart of a method 1200 for video processing in accordance with some embodiments of the present disclosure. The method 1200 may be implemented during a conversion between a current block of a video and a bitstream of the video. As shown in Fig. 12, the method 1200 starts at 1202 where motion information of the current block is determined based on motion information of a neighboring block of the current block and an IBC-MBVD mode. The neighboring block is coded with a reconstruction-reordered intra block copy (RRIBC) mode. By way of example rather than limitation, the motion information may comprise a motion vector, a reference index, a reference list, and/or the like. In some embodiments, the motion information of the neighboring block may be used as a base motion candidate for the IBC-MBVD mode, so as to generate the motion information of the current block.

[0083] In some embodiments, in the RRIBC mode, an adjustment may be applied on reconstruction samples of the neighboring block. By way of example rather than limitation, the adjustment may comprise reordering the reconstruction samples, flipping the reconstruction samples, shifting the reconstruction samples, rotating the reconstruction samples, transforming the reconstruction samples, and/or the like. It should be understood that the above examples are described merely for purpose of description. The scope of the present disclosure is not limited in this respect.

[0084] At 1204, the conversion is performed based on the motion information of the current block. In one example, the conversion may include encoding the current block into the bitstream. Alternatively or additionally, the conversion may include decoding the current block from the bitstream.

[0085] In view of the above, the motion information of the current block is determined based on motion information of an RRIBC-coded neighboring block and an IBC-MBVD mode. Compared with the conventional solution, the proposed method can better support the interaction between RRIBC mode and IBC-MBVD mode, and thus can advantageously improve the coding efficiency and coding quality.

[0086] In some embodiments, at 1202, the motion information of the neighboring block may be adjusted based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the motion information of the current block in the motion adjustment process. Moreover, the motion information of the current block may be generated based on the adjusted motion information and the IBC-MBVD mode. The adjusted motion information may be used as a base motion candidate for the IBC-MBVD mode, so as to generate the motion information of the current block.

[0087] In some alternative embodiments, at 1202, an intermediate motion information may be determined based on the motion information of the neighboring block and the IBC-MBVD mode. The motion information of the neighboring block may be used as a base motion candidate for the IBC-MBVD mode, so as to generate the intermediate motion information. Moreover, the motion information of the current block may be generated by adjusting the intermediate motion information based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the intermediate motion information in the motion adjustment process.

[0088] In some embodiments, the motion adjustment process may be dependent on a flip type of the current block, a flip type of the neighboring block, a position of the current block, a position of the neighboring block, a coordinate of the current block, a coordinate of the neighboring block, and/or the like.

[0089] In some embodiments, the adjustment may be performed along a single direction in the motion adjustment process. For example, if a flip type of the neighboring block is horizontal flip, the single direction may be a horizontal direction. Alternatively or additionally, if a flip type of the neighboring block is vertical flip, the single direction may be a vertical direction.

[0090] In some embodiments, a flip type of the current block may be determined independently from a flip type of the neighboring block. That is, the flip type of the neighboring block will not be considered for determining the flip type of the current block. In such a case, the current block may derive motion information (but not flip type) from the RRIBC coded neighboring block. For example, the flip type of the current block may always set equal to no-flip. In some alternative embodiments, a flip type of the current block may be determined based on a flip type of the neighboring block. In such a case, the current block may derive both the motion information and the flip type from the RRIBC coded neighboring block. For example, flip type of the current block may be set equal to the flip type of the neighboring block.

[0091] In some embodiments, the video may further comprise a further block different from the current block. The further neighboring block is coded with the RRIBC mode. Motion information of the further block may be determined based on the IBC-MBVD mode independently from motion information of a further neighboring block of the further block. That is, the motion information of the further neighboring block will not be considered for generating the motion information of the further block.

[0092] In some embodiments, the motion information of the further block may not be allowed to be used as a base motion candidate for the IBC-MBVD mode. Alternatively or additionally, the further block may not be allowed to be determined based on the further neighboring block. In some further embodiments, both the motion information of the further neighboring block and a flip type of the further neighboring block may not be allowed to be added into a motion candidate list for the further block. Alternatively or additionally, coding information of the further neighboring block may not be allowed to be added into a motion candidate list for the further block.

[0093] In some alternative embodiments, a flip type of the further block may be determined based on a flip type of the further neighboring block. In such a case, the further block may derive the flip type (but not the motion information) from the RRIBC coded further neighboring block. For example, the flip type of the further block may be set equal to the flip type of the further neighboring block.

[0094] In some embodiments, if the current block is coded with the RRIBC mode, a first set of MBVD offsets may be allowed to be added to a base motion candidate for the IBC-MBVD mode. If the current block is coded without the RRIBC mode, a second set of MBVD offsets may be allowed to be added to the base motion candidate. In some alternative or additional embodiments, if a base motion candidate for the IBC-MBVD mode is coded with the RRIBC mode, a first set of MBVD offsets may be allowed to be added to the base motion candidate. If the base motion candidate is coded without the RRIBC mode, a second set of MBVD offsets may be allowed to be added to the base motion candidate.

[0095] In some embodiments, the first set of MBVD offsets may be the same as the second set of MBVD offsets. In some alternative embodiments, the first set of MBVD offsets may be different from the second set of MBVD offsets. By way of example rather than limitation, the first set of MBVD offsets may be a subset of the second set of MBVD offsets. Alternatively, the second set of MBVD offsets may be a subset of the first set of

MBVD offsets.

[0096] In some embodiments, a set of MBVD offsets allowed to be added to a base motion candidate for the IBC-MBVD mode may be dependent on a flip type of the current block or a flip type of the base motion candidate. In one example, if the flip type of the current block is horizontal flip, the set of MBVD offsets may comprise a set of horizontal MBVD offsets. Additionally, a vertical component of each of the set of MBVD offsets may be equal to zero. If the flip type of the current block is vertical flip, the set of MBVD offsets may comprise a set of vertical MBVD offsets. Additionally, a horizontal component of each of the set of MBVD offsets may be equal to zero.

[0097] Additionally or alternatively, if the flip type of the base motion candidate is horizontal flip, the set of MBVD offsets may comprise a set of horizontal MBVD offsets. For example, a vertical component of each of the set of MBVD offsets may be equal to zero. If the flip type of the base motion candidate may be vertical flip, the set of MBVD offsets may comprise a set of vertical MBVD offsets. For example, a horizontal component of each of the set of MBVD offsets may be equal to zero.

[0098] In some embodiments, RRIBC-coded motion candidates for the IBC-MBVD mode and non-RRIBC-coded motion candidates for the IBC-MBVD mode may be group into a single set of motion candidates. Additionally, an index for a motion candidate for the IBC-MBVD mode may be coded based on the single set of motion candidates. Moreover, motion candidates in the single set of motion candidates may be ordered based on costs (such as template costs) of the motion candidates.

[0099] In some alternative embodiments, RRIBC-coded motion candidates for the IBC-MBVD mode may be group into a first set of motion candidates, and non-RRIBC-coded motion candidates for the IBC-MBVD mode may be group into a second set of motion candidates. Additionally, an index for a motion candidate for the IBC-MBVD mode may be coded based on the first set of motion candidates or the second set of motion candidates. Furthermore, motion candidates in the first set of motion candidates or the second set of motion candidates may be ordered based on template costs of the motion candidates.

[0100] As used herein, if a block is coded with RRIBC, the flip type of the block is not equal to no-flip, and the block may also be referred to as an RRIBC-coded block. If the current block is coded without RRIBC, the flip type of the current block is equal to no-flip, and the block may also be referred to as a non-RRIBC-coded block.

[0101] In some embodiments, whether to and/or how to apply the method may be indicated at a sequence level, a group of pictures level, a picture level, a slice level, a tile group level, or the like. In some embodiments, whether to and/or how to apply the method may be indicated in 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, a tile group header, or the like.

[0102] 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. In the method, motion information of a current block of the video is determined based on motion information of a neighboring block of the current block and an IBC-MBVD mode. The neighboring block is coded with an RRIBC mode. Moreover, the bitstream is generated based on the motion information of the current block.

[0103] According to still further embodiments of the present disclosure, a method for storing bitstream of a video is provided. In the method, motion information of a current block of the video is determined based on motion information of a neighboring block of the current block and an IBC-MBVD mode. The neighboring block is coded with an RRIBC mode. Moreover, the bitstream is generated based on the motion information of the current block, and the bitstream is stored in a non-transitory computer-readable recording medium.

[0104] Fig. 13 illustrates a flowchart of another method 1300 for video processing in accordance with some embodiments of the present disclosure. The method 1300 may be implemented during a conversion between a current block of a video and a bitstream of the video. As shown in Fig. 13, the method 1300 starts at 1302 where motion information of the current block is determined based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode. Both the current block and the neighboring block are coded with a RRIBC mode. By way of example rather than limitation, the motion information may comprise a motion vector, a reference index, a reference list, and/or the like. In some embodiments, the motion information of the current block may be the same as the motion information of the neighboring block. Additionally or alternatively, a flip type of the current block may be the same as a flip type of the neighboring block. In other words, the motion information and/or the flip type of the current block may be directly inherited from the RRIBC-coded neighboring block.

[0105] In some embodiments, in the RRIBC mode, an adjustment may be applied on reconstruction samples of the neighboring block or the current block. By way of example rather than limitation, the adjustment may comprise reordering the reconstruction samples, flipping the reconstruction samples, shifting the reconstruction samples, rotating the reconstruction samples, transforming the reconstruction samples, and/or the like. It should be understood that the above examples are described merely for purpose of description. The scope of the present disclosure is not limited in this respect.

[0106] At 1304, the conversion is performed based on the motion information of the current block. In one example, the conversion may include encoding the current block into the bitstream. Alternatively or additionally, the conversion may include decoding the current block from the bitstream.

[0107] In view of the above, the motion information of the RRIBC-coded current block is determined based on motion information of an RRIBC-coded neighboring block and an IBC-TM-MERGE mode. Compared with the conventional solution, the proposed method can better support the interaction between RRIBC mode and IBC-TM-MERGE mode, and thus can advantageously improve the coding efficiency and coding quality.

[0108] In some embodiments, at 1302, the motion information of the current block may be generated by adjusting the motion information of the neighboring block based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the motion information in the motion adjustment process.

[0109] Alternatively, at 1302, an intermediate motion information may be obtained by adjusting the motion information of the neighboring block based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the motion information in the motion adjustment process. Furthermore, the motion information of the current block may be generated by refining the intermediate motion information based on a template matching based motion refinement process.

[0110] In some alternative embodiments, at 1302, an intermediate motion information may be obtained by refining the motion information of the neighboring block based on a template matching based motion refinement process. Moreover, the motion information of the current block may be generated by adjusting the intermediate motion information based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the intermediate motion information in the motion adjustment process.

[0111] In some further embodiments, at 1302, the motion information of the current block may be generated by refining the motion information of the neighboring block based on a template matching based motion refinement process.

[0112] In some embodiments, the motion adjustment process may be dependent on a flip type of the current block, a flip type of the neighboring block, a position of the current block, a position of the neighboring block, a coordinate of the current block, or a coordinate of the neighboring block, and/or the like.

[0113] In some embodiments, the adjustment may be performed along a single direction in the motion adjustment process. For example, if a flip type of the neighboring block is horizontal flip, the single direction may be a horizontal direction. If a flip type of the neighboring block is vertical flip, the single direction may be a vertical direction.

[0114] In some embodiments, a set of motion vector (MV) offsets allowed to be used in the template matching based motion refinement process may be dependent on a flip type of the current block or a flip type of a motion candidate for the IBC-TM-MERGE mode. In the template matching based motion refinement process, the set of MV offsets may be used to refine the motion information (such as the motion vector) of the current block or an IBC-TM-MERGE candidate of the current block.

[0115] In some embodiments, if the flip type of the current block is horizontal flip, the set of MV offsets may comprise a set of horizontal MV offsets. For example, a vertical component of each of the set of MV offsets may be equal to zero. If the flip type of the motion candidate is horizontal flip, the set of MV offsets may comprise a set of horizontal MV offsets. For example, a vertical component of each of the set of MV offsets may be equal to zero.

[0116] Additionally or alternatively, if the flip type of the current block is vertical flip, the set of MV offsets may comprise a set of vertical MV offsets. For example, a horizontal component of each of the set of MV offsets may be equal to zero. If the flip type of the motion candidate is vertical flip, the set of MV offsets may comprise a set of vertical MV offsets. For example, a horizontal component of each of the set of MV offsets may be equal to zero.

[0117] In some embodiments, whether to and/or how to apply the method may be indicated at a sequence level, a group of pictures level, a picture level, a slice level, a tile group level, or the like. In some embodiments, whether to and/or how to apply the method may be indicated in 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, a tile group header, or the like.

[0118] 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. In the method, motion information of a current block of the video is determined based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode. The current block and the neighboring block are coded with an RRIBC mode. Moreover, the bitstream is generated based on the motion information of the current block.

[0119] According to still further embodiments of the present disclosure, a method for storing bitstream of a video is provided. In the method, motion information of a current block of the video is determined based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode. The current block and the neighboring block are coded with an RRIBC mode. Moreover, the bitstream is generated based on the motion information of the current block and is further stored in a non-transitory computer-readable recording medium.

[0120] Fig. 14 illustrates a flowchart of a further method 1400 for video processing in accordance with some embodiments of the present disclosure. The method 1400 may be implemented during a conversion between a current block of a video and a bitstream of the video. As shown in Fig. 14, the method 1400 starts at 1402 where a motion candidate for the current block is determined based on motion information of a neighboring block of the current block and an IBC-AMVP mode. The neighboring block is coded with an RRIBC mode. In some embodiments, the current block may be coded with the RRIBC mode. Alternatively, the current block may be coded without the RRIBC mode.

[0121] In some embodiments, in the RRIBC mode, an adjustment may be applied on reconstruction samples of the neighboring block. By way of example rather than limitation, the adjustment may comprise reordering the reconstruction samples, flipping the reconstruction samples, shifting the reconstruction samples, rotating the reconstruction samples, transforming the reconstruction samples, and/or the like. It should be understood that the above examples are described merely for purpose of description. The scope of the present disclosure is not limited in this respect.

[0122] At 1404, motion information of the current block is determined based on the motion candidate. In some embodiments, the motion information of the current block may be generated by adjusting a motion vector of the motion candidate based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the motion vector in the motion adjustment process.

[0123] Alternatively, an intermediate motion vector may be obtained by adjusting a motion vector of the motion candidate based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the motion vector in the motion adjustment process. Moreover, the motion information of the current block may be generated by refining the intermediate motion vector based on a template matching based motion refinement process.

[0124] In some alternative embodiments, an intermediate motion vector may be obtained by refining a motion vector of the motion candidate based on a template matching based motion refinement process. Furthermore, the motion information of the current block may be generated by adjusting the intermediate motion vector based on a motion adjustment process. By way of example rather than limitation, a motion shift may be added to the intermediate motion vector in the motion adjustment process.

[0125] In some further embodiments, the motion information of the current block may be generated by refining a motion vector of the motion candidate based on a template matching based motion refinement process. It should be understood that the above examples are described merely for purpose of description. The scope of the present disclosure is not limited in this respect.

[0126] At 1406, the conversion is performed based on the motion information of the current block. In one example, the conversion may include encoding the current block into the bitstream. Alternatively or additionally, the conversion may include decoding the current block from the bitstream.

[0127] In view of the above, the motion information of the current block is determined based on motion information of an RRIBC-coded neighboring block and an IBC-AMVP mode. Compared with the conventional solution, the proposed method can better support the interaction between RRIBC mode and IBC-AMVP mode, and thus can advantageously improve the coding efficiency and coding quality.

[0128] In some embodiments, the motion adjustment process may be dependent on a flip type of the current block, a flip type of the neighboring block, a position of the current block, a position of the neighboring block, a coordinate of the current block, a coordinate of the neighboring block, and/or the like.

[0129] In some embodiments, the adjustment may be performed along a single direction in the motion adjustment process. For example, if a flip type of the neighboring block is horizontal flip, the single direction may be a horizontal direction. If a flip type of the neighboring block is vertical flip, the single direction may be a vertical direction.

[0130] In some embodiments, a set of motion vector (MV) offsets allowed to be used in the template matching based motion refinement process may be dependent on a flip type of the current block or a flip type of the motion candidate. In the template matching based motion refinement process, the set of MV offsets may be used to refine the motion information (such as the motion vector) of the current block or an IBC-TM-AMVP candidate of the current block.

[0131] In some embodiments, if the flip type of the current block is horizontal flip, the set of MV offsets may comprise a set of horizontal MV offsets. For example, a vertical component of each of the set of MV offsets may be equal to zero. If the flip type of the motion candidate is horizontal flip, the set of MV offsets may comprise a set of horizontal MV offsets. For example, a vertical component of each of the set of MV offsets may be equal to zero.

[0132] Additionally or alternatively, if the flip type of the current block is vertical flip, the set of MV offsets may comprise a set of vertical MV offsets. For example, a horizontal component of each of the set of MV offsets may be equal to zero. If the flip type of the motion candidate is vertical flip, the set of MV offsets may comprise a set of vertical MV offsets. For example, a horizontal component of each of the set of MV offsets may be equal to zero.

[0133] In some embodiments, whether to and/or how to apply the method may be indicated at a sequence level, a group of pictures level, a picture level, a slice level, a tile group level, or the like. In some embodiments, whether to and/or how to apply the method may be indicated in 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, a tile group header, or the like.

[0134] 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. In the method, a motion candidate for a current block of the video is determined based on motion information of a neighboring block of the current block and an IBC-AMVP mode. The neighboring block is coded with an RRIBC mode. Motion information of the current block is determined based on the motion candidate. Moreover, the bitstream is generated based on the motion information of the current block.

[0135] According to still further embodiments of the present disclosure, a method for storing bitstream of a video is provided. In the method, a motion candidate for a current block of the video is determined based on motion information of a neighboring block of the current block and an IBC-AMVP mode. The neighboring block is coded with an RRIBC mode. Motion information of the current block is determined based on the motion candidate. Moreover, the bitstream is generated based on the motion information of the current block, and the bitstream is further stored in a non-transitory computer-readable recording medium.

[0136] 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.

[0137] Clause 1. A method for video processing, comprising: determining, for a conversion between a current block of a video and a bitstream of the video, motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy merge mode with block vector difference (IBC-MBVD) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and performing the conversion based on the motion information of the current block.

[0138] Clause 2. The method of clause 1, wherein the motion information of the neighboring block is used as a base motion candidate for the IBC-MBVD mode.

[0139] Clause 3. The method of clause 1, wherein determining the motion information of the current block comprises: adjusting the motion information of the neighboring block based on a motion adjustment process; and generating the motion information of the current block based on the adjusted motion information and the IBC-MBVD mode, the adjusted motion information being used as a base motion candidate for the IBC-MBVD mode.

[0140] Clause 4. The method of clause 3, wherein a motion shift is added to the motion information of the current block in the motion adjustment process.

[0141] Clause 5. The method of clause 1, wherein determining the motion information of the current block comprises: determining an intermediate motion information based on the motion information of the neighboring block and the IBC-MBVD mode, the motion information of the neighboring block being used as a base motion candidate for the IBC-MBVD mode; and generating the motion information of the current block by adjusting the intermediate motion information based on a motion adjustment process.

[0142] Clause 6. The method of clause 5, wherein a motion shift is added to the intermediate motion information in the motion adjustment process.

[0143] Clause 7. The method of any of clauses 3-6, wherein the motion adjustment process is dependent on at least one of the following: a flip type of the current block, a flip type of the neighboring block, a position of the current block, a position of the neighboring block, a coordinate of the current block, or a coordinate of the neighboring block.

[0144] Clause 8. The method of any of clauses 3-7, wherein the adjustment is performed along a single direction in the motion adjustment process.

[0145] Clause 9. The method of clause 8, wherein if a flip type of the neighboring block is horizontal flip, the single direction is a horizontal direction.

[0146] Clause 10. The method of clause 8, wherein if a flip type of the neighboring block is vertical flip, the single direction is a vertical direction.

[0147] Clause 11. The method of any of clauses 1-10, wherein a flip type of the current block is determined independently from a flip type of the neighboring block.

[0148] Clause 12. The method of clause 11, wherein the flip type of the current block is no-flip.

[0149] Clause 13. The method of any of clauses 1-10, wherein a flip type of the current block is determined based on a flip type of the neighboring block.

[0150] Clause 14. The method of clause 13, wherein flip type of the current block is the same as the flip type of the neighboring block.

[0151] Clause 15. The method of any of clauses 1-14, wherein the video further comprises a further block different from the current block, motion information of the further block is determined based on the IBC-MBVD mode independently from motion information of a further neighboring block of the further block, and the further neighboring block is coded with the RRIBC mode.

[0152] Clause 16. The method of clause 15, wherein the motion information of the further block is not allowed to be used as a base motion candidate for the IBC-MBVD mode.

[0153] Clause 17. The method of any of clauses 15-16, wherein the further block is not allowed to be determined based on the further neighboring block.

[0154] Clause 18. The method of any of clauses 15-17, wherein the motion information of the further neighboring block and a flip type of the further neighboring block are not allowed to be added into a motion candidate list for the further block.

[0155] Clause 19. The method of any of clauses 15-18, wherein coding information of the further neighboring block is not allowed to be added into a motion candidate list for the further block.

[0156] Clause 20. The method of any of clauses 15-16, wherein a flip type of the further block is determined based on a flip type of the further neighboring block.

[0157] Clause 21. The method of any of clauses 1-20, wherein if the current block is coded with the RRIBC mode, a first set of MBVD offsets is allowed to be added to a base motion candidate for the IBC-MBVD mode, if the current block is coded without the

RRIBC mode, a second set of MBVD offsets is allowed to be added to the base motion candidate.

[0158] Clause 22. The method of any of clauses 1-20, wherein if a base motion candidate for the IBC-MBVD mode is coded with the RRIBC mode, a first set of MBVD offsets is allowed to be added to the base motion candidate, if the base motion candidate is coded without the RRIBC mode, a second set of MBVD offsets is allowed to be added to the base motion candidate.

[0159] Clause 23. The method of any of clauses 21-22, wherein the first set of MBVD offsets is different from the second set of MBVD offsets.

[0160] Clause 24. The method of any of clauses 21-22, wherein the first set of MBVD offsets is the same as the second set of MBVD offsets.

[0161] Clause 25. The method of any of clauses 21-23, wherein the first set of MBVD offsets is a subset of the second set of MBVD offsets.

[0162] Clause 26. The method of any of clauses 1-20, wherein a set of MBVD offsets allowed to be added to a base motion candidate for the IBC-MBVD mode is dependent on a flip type of the current block or a flip type of the base motion candidate.

[0163] Clause 27. The method of clause 26, wherein if the flip type of the current block is horizontal flip, the set of MBVD offsets comprises a set of horizontal MBVD offsets.

[0164] Clause 28. The method of clause 27, wherein a vertical component of each of the set of MBVD offsets is equal to zero.

[0165] Clause 29. The method of clause 26, wherein if the flip type of the current block is vertical flip, the set of MBVD offsets comprises a set of vertical MBVD offsets.

[0166] Clause 30. The method of clause 29, wherein a horizontal component of each of the set of MBVD offsets is equal to zero.

[0167] Clause 31. The method of clause 26, wherein if the flip type of the base motion candidate is horizontal flip, the set of MBVD offsets comprises a set of horizontal MBVD offsets.

[0168] Clause 32. The method of clause 31, wherein a vertical component of each of the set of MBVD offsets is equal to zero.

[0169] Clause 33. The method of clause 26, wherein if the flip type of the base motion candidate is vertical flip, the set of MBVD offsets comprises a set of vertical MBVD offsets.

[0170] Clause 34. The method of clause 33, wherein a horizontal component of each of the set of MBVD offsets is equal to zero.

[0171] Clause 35. The method of any of clauses 1-34, wherein RRIBC-coded motion candidates for the IBC-MBVD mode and non-RRIBC-coded motion candidates for the IBC-MBVD mode are group into a single set of motion candidates.

[0172] Clause 36. The method of clause 35, wherein an index for a motion candidate for the IBC-MBVD mode is coded based on the single set of motion candidates.

[0173] Clause 37. The method of clause 35, wherein motion candidates in the single set of motion candidates are ordered based on template costs of the motion candidates.

[0174] Clause 38. The method of any of clauses 1-34, wherein RRIBC-coded motion candidates for the IBC-MBVD mode are group into a first set of motion candidates, and non-RRIBC-coded motion candidates for the IBC-MBVD mode are group into a second set of motion candidates.

[0175] Clause 39. The method of clause 38, wherein an index for a motion candidate for the IBC-MBVD mode is coded based on the first set of motion candidates or the second set of motion candidates.

[0176] Clause 40. The method of clause 38, wherein motion candidates in the first set of motion candidates or the second set of motion candidates are ordered based on template costs of the motion candidates.

[0177] Clause 41. The method of any of clauses 1-40, wherein if the current block is coded with RRIBC, the flip type of the current block is not equal to no-flip, if the current block is coded without RRIBC, the flip type of the current block is equal to no-flip.

[0178] Clause 42. A method for video processing, comprising: determining, for a conversion between a current block of a video and a bitstream of the video, motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy template matching merge (IBC-TM-MERGE) mode, the current block and the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and performing the conversion based on the motion information of the current block.

[0179] Clause 43. The method of clause 42, wherein the motion information of the current block is the same as the motion information of the neighboring block.

[0180] Clause 44. The method of any of clauses 42-43, wherein a flip type of the current block is the same as a flip type of the neighboring block.

[0181] Clause 45. The method of clause 42, wherein determining the motion information of the current block comprises: generating the motion information of the current block by adjusting the motion information of the neighboring block based on a motion adjustment process.

[0182] Clause 46. The method of clause 42, wherein determining the motion information of the current block comprises: obtaining an intermediate motion information by adjusting the motion information of the neighboring block based on a motion adjustment process; and generating the motion information of the current block by refining the intermediate motion information based on a template matching based motion refinement process.

[0183] Clause 47. The method of clause 42, wherein determining the motion information of the current block comprises: obtaining an intermediate motion information by refining the motion information of the neighboring block based on a template matching based motion refinement process; and generating the motion information of the current block by adjusting the intermediate motion information based on a motion adjustment process.

[0184] Clause 48. The method of clause 42, wherein determining the motion information of the current block comprises: generating the motion information of the current block by refining the motion information of the neighboring block based on a template matching based motion refinement process.

[0185] Clause 49. The method of any of clauses 45-47, wherein the motion adjustment process is dependent on at least one of the following: a flip type of the current block, a flip type of the neighboring block, a position of the current block, a position of the neighboring block, a coordinate of the current block, or a coordinate of the neighboring block.

[0186] Clause 50. The method of any of clauses 45-47 and 49, wherein the adjustment is performed along a single direction in the motion adjustment process.

[0187] Clause 51. The method of clause 50, wherein if a flip type of the neighboring block is horizontal flip, the single direction is a horizontal direction.

[0188] Clause 52. The method of clause 50, wherein if a flip type of the neighboring block is vertical flip, the single direction is a vertical direction.

[0189] Clause 53. The method of any of clauses 46-48, wherein a set of motion vector (MV) offsets allowed to be used in the template matching based motion refinement process is dependent on a flip type of the current block or a flip type of a motion candidate for the IBC-TM-MERGE mode.

[0190] Clause 54. The method of clause 53, wherein if the flip type of the current block is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

[0191] Clause 55. The method of clause 54, wherein a vertical component of each of the set of MV offsets is equal to zero.

[0192] Clause 56. The method of clause 53, wherein if the flip type of the motion candidate is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

[0193] Clause 57. The method of clause 56, wherein a vertical component of each of the set of MV offsets is equal to zero.

[0194] Clause 58. The method of clause 53, wherein if the flip type of the current block is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

[0195] Clause 59. The method of clause 58, wherein a horizontal component of each of the set of MV offsets is equal to zero.

[0196] Clause 60. The method of clause 53, wherein if the flip type of the motion candidate is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

[0197] Clause 61. The method of clause 60, wherein a horizontal component of each of the set of MV offsets is equal to zero.

[0198] Clause 62. A method for video processing, comprising: determining, for a conversion between a current block of a video and a bitstream of the video, a motion candidate for the current block based on motion information of a neighboring block of the current block and an intra block copy advanced motion vector prediction (IBC-AMVP) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; determining motion information of the current block based on the motion candidate; and performing the conversion based on the motion information of the current block.

[0199] Clause 63. The method of clause 62, wherein determining the motion information of the current block comprises: generating the motion information of the current block by adjusting a motion vector of the motion candidate based on a motion adjustment process.

[0200] Clause 64. The method of clause 62, wherein determining the motion information of the current block comprises: obtaining an intermediate motion vector by adjusting a motion vector of the motion candidate based on a motion adjustment process; and generating the motion information of the current block by refining the intermediate motion vector based on a template matching based motion refinement process.

[0201] Clause 65. The method of clause 62, wherein determining the motion information of the current block comprises: obtaining an intermediate motion vector by refining a motion vector of the motion candidate based on a template matching based motion refinement process; and generating the motion information of the current block by adjusting the intermediate motion vector based on a motion adjustment process.

[0202] Clause 66. The method of clause 62, wherein determining the motion information of the current block comprises: generating the motion information of the current block by refining a motion vector of the motion candidate based on a template matching based motion refinement process.

[0203] Clause 67. The method of any of clauses 62-66, wherein the current block is coded without the RRIBC mode.

[0204] Clause 68. The method of any of clauses 63-65, wherein the motion adjustment process is dependent on at least one of the following: a flip type of the current block, a flip type of the neighboring block, a position of the current block, a position of the neighboring block, a coordinate of the current block, or a coordinate of the neighboring block.

[0205] Clause 69. The method of any of clauses 63-65 and 68, wherein the adjustment is performed along a single direction in the motion adjustment process.

[0206] Clause 70. The method of clause 69, wherein if a flip type of the neighboring block is horizontal flip, the single direction is a horizontal direction.

[0207] Clause 71. The method of clause 69, wherein if a flip type of the neighboring block is vertical flip, the single direction is a vertical direction.

[0208] Clause 72. The method of any of clauses 64-56, wherein a set of motion vector (MV) offsets allowed to be used in the template matching based motion refinement process is dependent on a flip type of the current block or a flip type of the motion candidate.

[0209] Clause 73. The method of clause 72, wherein if the flip type of the current block is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

[0210] Clause 74. The method of clause 73, wherein a vertical component of each of the set of MV offsets is equal to zero.

[0211] Clause 75. The method of clause 72, wherein if the flip type of the motion candidate is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

[0212] Clause 76. The method of clause 75, wherein a vertical component of each of the set of MV offsets is equal to zero.

[0213] Clause 77. The method of clause 72, wherein if the flip type of the current block is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

[0214] Clause 78. The method of clause 77, wherein a horizontal component of each of the set of MV offsets is equal to zero.

[0215] Clause 79. The method of clause 72, wherein if the flip type of the motion candidate is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

[0216] Clause 80. The method of clause 79, wherein a horizontal component of each of the set of MV offsets is equal to zero.

[0217] Clause 81. The method of any of clauses 1-80, wherein an adjustment is applied on reconstruction samples of the neighboring block in the RRIBC mode.

[0218] Clause 82. The method of clause 81, wherein the adjustment comprises at least one of the following: reordering the reconstruction samples, flipping the reconstruction samples, shifting the reconstruction samples, rotating the reconstruction samples, or transforming the reconstruction samples.

[0219] Clause 83. The method of any of clauses 1-82, wherein the current block is one of the following: a color component, a sub-picture, a slice, a tile, a coding tree unit (CTU), a CTU row, groups of CTU, a coding unit (CU), a prediction unit (PU), a transform unit (TU), a coding tree block (CTB), a coding block (CB), a prediction block (PB), a transform block (TB), a sub-block of a video block, or a sub-region within a video block.

[0220] Clause 84. The method of any of clauses 1-82, wherein whether to and/or how to apply the method is indicated at one of the following: a sequence level, a group of pictures level, a picture level, a slice level, or a tile group level.

[0221] Clause 85. The method of any of clauses 1-82, wherein whether to and/or how to apply the method is indicated in one of the following: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a dependency parameter set (DPS), a decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter sets (APS), a slice header, or a tile group header.

[0222] Clause 86. The method of any of clauses 1-85, wherein the conversion includes encoding the current block into the bitstream.

[0223] Clause 87. The method of any of clauses 1-85, wherein the conversion includes decoding the current block from the bitstream.

[0224] Clause 88. 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 1-87.

[0225] Clause 89. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of clauses 1-87.

[0226] Clause 90. 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: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-MBVD mode, the neighboring block being coded with an RRIBC mode;

and generating the bitstream based on the motion information of the current block.

[0227] Clause 91. A method for storing a bitstream of a video, comprising: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-MBVD mode, the neighboring block being coded with an RRIBC mode; generating the bitstream based on the motion information of the current block; storing the bitstream in a non-transitory computer-readable recording medium.

[0228] Clause 92. 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: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode, the current block and the neighboring block being coded with an RRIBC mode; and generating the bitstream based on the motion information of the current block.

[0229] Clause 93. A method for storing a bitstream of a video, comprising: determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode, the current block and the neighboring block being coded with an RRIBC mode; generating the bitstream based on the motion information of the current block; and storing the bitstream in a non-transitory computer-readable recording medium.

[0230] Clause 94. 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: determining a motion candidate for a current block of the video based on motion information of a neighboring block of the current block and an IBC-AMVP mode, the neighboring block being coded with an RRIBC mode; determining motion information of the current block based on the motion candidate; and generating the bitstream based on the motion information of the current block.

[0231] Clause 95. A method for storing a bitstream of a video, comprising: determining a motion candidate for a current block of the video based on motion information of a neighboring block of the current block and an IBC-AMVP mode, the neighboring block being coded with an RRIBC mode; determining motion information of the current block based on the motion candidate; generating the bitstream based on the motion information of the current block; and storing the bitstream in a non-transitory computer-readable recording medium.

Example Device

[0232] Fig. 15 illustrates a block diagram of a computing device 1500 in which various embodiments of the present disclosure can be implemented. The computing device 1500 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).

[0233] It would be appreciated that the computing device 1500 shown in Fig. 15 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.

[0234] As shown in Fig. 15, the computing device 1500 includes a general-purpose computing device 1500. The computing device 1500 may at least comprise one or more processors or processing units 1510, a memory 1520, a storage unit 1530, one or more communication units 1540, one or more input devices 1550, and one or more output devices 1560.

[0235] In some embodiments, the computing device 1500 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 1500 can support any type of interface to a user (such as “wearable” circuitry and the like).

[0236] The processing unit 1510 may be a physical or virtual processor and can implement various processes based on programs stored in the memory 1520. 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 1500. The processing unit 1510 may also be referred to as a central processing unit (CPU), a microprocessor, a controller or a microcontroller.

[0237] The computing device 1500 typically includes various computer storage medium. Such medium can be any medium accessible by the computing device 1500, including, but not limited to, volatile and non-volatile medium, or detachable and non-detachable medium. The memory 1520 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 1530 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 1500.

[0238] The computing device 1500 may further include additional detachable/non-detachable, volatile/non-volatile memory medium. Although not shown in Fig. 15, 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.

[0239] The communication unit 1540 communicates with a further computing device via the communication medium. In addition, the functions of the components in the computing device 1500 can be implemented by a single computing cluster or multiple computing machines that can communicate via communication connections. Therefore, the computing device 1500 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.

[0240] The input device 1550 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 1560 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 1540, the computing device 1500 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 1500, or any devices (such as a network card, a modem and the like) enabling the computing device 1500 to communicate with one or more other computing devices, if required. Such communication can be performed via input/output (I/O) interfaces (not shown).

[0241] In some embodiments, instead of being integrated in a single device, some or all components of the computing device 1500 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.

[0242] The computing device 1500 may be used to implement video encoding/decoding in embodiments of the present disclosure. The memory 1520 may include one or more video coding modules 1525 having one or more program instructions. These modules are accessible and executable by the processing unit 1510 to perform the functionalities of the various embodiments described herein.

[0243] In the example embodiments of performing video encoding, the input device 1550 may receive video data as an input 1570 to be encoded. The video data may be processed, for example, by the video coding module 1525, to generate an encoded bitstream. The encoded bitstream may be provided via the output device 1560 as an output 1580.

[0244] In the example embodiments of performing video decoding, the input device 1550 may receive an encoded bitstream as the input 1570. The encoded bitstream may be processed, for example, by the video coding module 1525, to generate decoded video data. The decoded video data may be provided via the output device 1560 as the output 1580.

[0245] 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:

determining, for a conversion between a current block of a video and a bitstream of the video, motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy merge mode with block vector difference (IBC-MBVD) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and

performing the conversion based on the motion information of the current block.

2. The method of claim 1, wherein the motion information of the neighboring block is used as a base motion candidate for the IBC-MBVD mode.

3. The method of claim 1, wherein determining the motion information of the current block comprises:

adjusting the motion information of the neighboring block based on a motion adjustment process; and

generating the motion information of the current block based on the adjusted motion information and the IBC-MBVD mode, the adjusted motion information being used as a base motion candidate for the IBC-MBVD mode.

4. The method of claim 3, wherein a motion shift is added to the motion information of the current block in the motion adjustment process.

5. The method of claim 1, wherein determining the motion information of the current block comprises:

determining an intermediate motion information based on the motion information of the neighboring block and the IBC-MBVD mode, the motion information of the neighboring block being used as a base motion candidate for the IBC-MBVD mode; and

generating the motion information of the current block by adjusting the intermediate motion information based on a motion adjustment process.

6. The method of claim 5, wherein a motion shift is added to the intermediate motion information in the motion adjustment process.

7. The method of any of claims 3-6, wherein the motion adjustment process is dependent on at least one of the following:

a flip type of the current block,
a flip type of the neighboring block,
a position of the current block,
a position of the neighboring block,
a coordinate of the current block, or
a coordinate of the neighboring block.

8. The method of any of claims 3-7, wherein the adjustment is performed along a single direction in the motion adjustment process.

9. The method of claim 8, wherein if a flip type of the neighboring block is horizontal flip, the single direction is a horizontal direction.

10. The method of claim 8, wherein if a flip type of the neighboring block is vertical flip, the single direction is a vertical direction.

11. The method of any of claims 1-10, wherein a flip type of the current block is determined independently from a flip type of the neighboring block.

12. The method of claim 11, wherein the flip type of the current block is no-flip.

13. The method of any of claims 1-10, wherein a flip type of the current block is determined based on a flip type of the neighboring block.

14. The method of claim 13, wherein flip type of the current block is the same as the flip type of the neighboring block.

15. The method of any of claims 1-14, wherein the video further comprises a further block different from the current block, motion information of the further block is determined based on the IBC-MBVD mode independently from motion information of a further

neighboring block of the further block, and the further neighboring block is coded with the RRIBC mode.

16. The method of claim 15, wherein the motion information of the further block is not allowed to be used as a base motion candidate for the IBC-MBVD mode.

17. The method of any of claims 15-16, wherein the further block is not allowed to be determined based on the further neighboring block.

18. The method of any of claims 15-17, wherein the motion information of the further neighboring block and a flip type of the further neighboring block are not allowed to be added into a motion candidate list for the further block.

19. The method of any of claims 15-18, wherein coding information of the further neighboring block is not allowed to be added into a motion candidate list for the further block.

20. The method of any of claims 15-16, wherein a flip type of the further block is determined based on a flip type of the further neighboring block.

21. The method of any of claims 1-20, wherein if the current block is coded with the RRIBC mode, a first set of MBVD offsets is allowed to be added to a base motion candidate for the IBC-MBVD mode,

if the current block is coded without the RRIBC mode, a second set of MBVD offsets is allowed to be added to the base motion candidate.

22. The method of any of claims 1-20, wherein if a base motion candidate for the IBC-MBVD mode is coded with the RRIBC mode, a first set of MBVD offsets is allowed to be added to the base motion candidate,

if the base motion candidate is coded without the RRIBC mode, a second set of MBVD offsets is allowed to be added to the base motion candidate.

23. The method of any of claims 21-22, wherein the first set of MBVD offsets is different from the second set of MBVD offsets.

24. The method of any of claims 21-22, wherein the first set of MBVD offsets is the same as the second set of MBVD offsets.

25. The method of any of claims 21-23, wherein the first set of MBVD offsets is a subset of the second set of MBVD offsets.

26. The method of any of claims 1-20, wherein a set of MBVD offsets allowed to be added to a base motion candidate for the IBC-MBVD mode is dependent on a flip type of the current block or a flip type of the base motion candidate.

27. The method of claim 26, wherein if the flip type of the current block is horizontal flip, the set of MBVD offsets comprises a set of horizontal MBVD offsets.

28. The method of claim 27, wherein a vertical component of each of the set of MBVD offsets is equal to zero.

29. The method of claim 26, wherein if the flip type of the current block is vertical flip, the set of MBVD offsets comprises a set of vertical MBVD offsets.

30. The method of claim 29, wherein a horizontal component of each of the set of MBVD offsets is equal to zero.

31. The method of claim 26, wherein if the flip type of the base motion candidate is horizontal flip, the set of MBVD offsets comprises a set of horizontal MBVD offsets.

32. The method of claim 31, wherein a vertical component of each of the set of MBVD offsets is equal to zero.

33. The method of claim 26, wherein if the flip type of the base motion candidate is vertical flip, the set of MBVD offsets comprises a set of vertical MBVD offsets.

34. The method of claim 33, wherein a horizontal component of each of the set of MBVD offsets is equal to zero.

35. The method of any of claims 1-34, wherein RRIBC-coded motion candidates for the IBC-MBVD mode and non-RRIBC-coded motion candidates for the IBC-MBVD mode are group into a single set of motion candidates.

36. The method of claim 35, wherein an index for a motion candidate for the IBC-MBVD mode is coded based on the single set of motion candidates.

37. The method of claim 35, wherein motion candidates in the single set of motion candidates are ordered based on template costs of the motion candidates.

38. The method of any of claims 1-34, wherein RRIBC-coded motion candidates for the IBC-MBVD mode are group into a first set of motion candidates, and non-RRIBC-coded motion candidates for the IBC-MBVD mode are group into a second set of motion candidates.

39. The method of claim 38, wherein an index for a motion candidate for the IBC-MBVD mode is coded based on the first set of motion candidates or the second set of motion candidates.

40. The method of claim 38, wherein motion candidates in the first set of motion candidates or the second set of motion candidates are ordered based on template costs of the motion candidates.

41. The method of any of claims 1-40, wherein if the current block is coded with RRIBC, the flip type of the current block is not equal to no-flip,

if the current block is coded without RRIBC, the flip type of the current block is equal to no-flip.

42. A method for video processing, comprising:

determining, for a conversion between a current block of a video and a bitstream of the video, motion information of the current block based on motion information of a neighboring block of the current block and an intra block copy template matching merge (IBC-TM-MERGE) mode, the current block and the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode; and

performing the conversion based on the motion information of the current block.

43. The method of claim 42, wherein the motion information of the current block is the same as the motion information of the neighboring block.

44. The method of any of claims 42-43, wherein a flip type of the current block is the same as a flip type of the neighboring block.

45. The method of claim 42, wherein determining the motion information of the current block comprises:

generating the motion information of the current block by adjusting the motion information of the neighboring block based on a motion adjustment process.

46. The method of claim 42, wherein determining the motion information of the current block comprises:

obtaining an intermediate motion information by adjusting the motion information of the neighboring block based on a motion adjustment process; and

generating the motion information of the current block by refining the intermediate motion information based on a template matching based motion refinement process.

47. The method of claim 42, wherein determining the motion information of the current block comprises:

obtaining an intermediate motion information by refining the motion information of the neighboring block based on a template matching based motion refinement process; and

generating the motion information of the current block by adjusting the intermediate motion information based on a motion adjustment process.

48. The method of claim 42, wherein determining the motion information of the current block comprises:

generating the motion information of the current block by refining the motion information of the neighboring block based on a template matching based motion refinement process.

49. The method of any of claims 45-47, wherein the motion adjustment process is dependent on at least one of the following:

a flip type of the current block,
a flip type of the neighboring block,
a position of the current block,
a position of the neighboring block,
a coordinate of the current block, or
a coordinate of the neighboring block.

50. The method of any of claims 45-47 and 49, wherein the adjustment is performed along a single direction in the motion adjustment process.

51. The method of claim 50, wherein if a flip type of the neighboring block is horizontal flip, the single direction is a horizontal direction.

52. The method of claim 50, wherein if a flip type of the neighboring block is vertical flip, the single direction is a vertical direction.

53. The method of any of claims 46-48, wherein a set of motion vector (MV) offsets allowed to be used in the template matching based motion refinement process is dependent on a flip type of the current block or a flip type of a motion candidate for the IBC-TM-MERGE mode.

54. The method of claim 53, wherein if the flip type of the current block is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

55. The method of claim 54, wherein a vertical component of each of the set of MV offsets is equal to zero.

56. The method of claim 53, wherein if the flip type of the motion candidate is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

57. The method of claim 56, wherein a vertical component of each of the set of MV offsets is equal to zero.

58. The method of claim 53, wherein if the flip type of the current block is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

59. The method of claim 58, wherein a horizontal component of each of the set of MV offsets is equal to zero.

60. The method of claim 53, wherein if the flip type of the motion candidate is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

61. The method of claim 60, wherein a horizontal component of each of the set of MV offsets is equal to zero.

62. A method for video processing, comprising:

determining, for a conversion between a current block of a video and a bitstream of the video, a motion candidate for the current block based on motion information of a neighboring block of the current block and an intra block copy advanced motion vector prediction (IBC-AMVP) mode, the neighboring block being coded with a reconstruction-reordered intra block copy (RRIBC) mode;

determining motion information of the current block based on the motion candidate; and

performing the conversion based on the motion information of the current block.

63. The method of claim 62, wherein determining the motion information of the current block comprises:

generating the motion information of the current block by adjusting a motion vector of the motion candidate based on a motion adjustment process.

64. The method of claim 62, wherein determining the motion information of the current block comprises:

obtaining an intermediate motion vector by adjusting a motion vector of the motion candidate based on a motion adjustment process; and

generating the motion information of the current block by refining the intermediate motion vector based on a template matching based motion refinement process.

65. The method of claim 62, wherein determining the motion information of the current block comprises:

obtaining an intermediate motion vector by refining a motion vector of the motion candidate based on a template matching based motion refinement process; and

generating the motion information of the current block by adjusting the intermediate motion vector based on a motion adjustment process.

66. The method of claim 62, wherein determining the motion information of the current block comprises:

generating the motion information of the current block by refining a motion vector of the motion candidate based on a template matching based motion refinement process.

67. The method of any of claims 62-66, wherein the current block is coded without the RRIBC mode.

68. The method of any of claims 63-65, wherein the motion adjustment process is dependent on at least one of the following:

a flip type of the current block,
a flip type of the neighboring block,
a position of the current block,
a position of the neighboring block,
a coordinate of the current block, or
a coordinate of the neighboring block.

69. The method of any of claims 63-65 and 68, wherein the adjustment is performed along a single direction in the motion adjustment process.

70. The method of claim 69, wherein if a flip type of the neighboring block is horizontal flip, the single direction is a horizontal direction.

71. The method of claim 69, wherein if a flip type of the neighboring block is vertical flip, the single direction is a vertical direction.

72. The method of any of claims 64-56, wherein a set of motion vector (MV) offsets allowed to be used in the template matching based motion refinement process is dependent on a flip type of the current block or a flip type of the motion candidate.

73. The method of claim 72, wherein if the flip type of the current block is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

74. The method of claim 73, wherein a vertical component of each of the set of MV offsets is equal to zero.

75. The method of claim 72, wherein if the flip type of the motion candidate is horizontal flip, the set of MV offsets comprises a set of horizontal MV offsets.

76. The method of claim 75, wherein a vertical component of each of the set of MV offsets is equal to zero.

77. The method of claim 72, wherein if the flip type of the current block is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

78. The method of claim 77, wherein a horizontal component of each of the set of MV offsets is equal to zero.

79. The method of claim 72, wherein if the flip type of the motion candidate is vertical flip, the set of MV offsets comprises a set of vertical MV offsets.

80. The method of claim 79, wherein a horizontal component of each of the set of MV offsets is equal to zero.

81. The method of any of claims 1-80, wherein an adjustment is applied on reconstruction samples of the neighboring block in the RRIBC mode.

82. The method of claim 81, wherein the adjustment comprises at least one of the following:

reordering the reconstruction samples,

flipping the reconstruction samples,
shifting the reconstruction samples,
rotating the reconstruction samples, or
transforming the reconstruction samples.

83. The method of any of claims 1-82, wherein the current block is one of the following:
a color component,
a sub-picture,
a slice,
a tile,
a coding tree unit (CTU),
a CTU row,
groups of CTU,
a coding unit (CU),
a prediction unit (PU),
a transform unit (TU),
a coding tree block (CTB),
a coding block (CB),
a prediction block (PB),
a transform block (TB),
a sub-block of a video block, or
a sub-region within a video block.

84. The method of any of claims 1-82, wherein whether to and/or how to apply the method is indicated at one of the following:
a sequence level,
a group of pictures level,
a picture level,
a slice level, or
a tile group level.

85. The method of any of claims 1-82, wherein whether to and/or how to apply the method is indicated in one of the following:
a sequence header,

a picture header,
a sequence parameter set (SPS),
a video parameter set (VPS),
a dependency parameter set (DPS),
a decoding capability information (DCI),
a picture parameter set (PPS),
an adaptation parameter sets (APS),
a slice header, or
a tile group header.

86. The method of any of claims 1-85, wherein the conversion includes encoding the current block into the bitstream.

87. The method of any of claims 1-85, wherein the conversion includes decoding the current block from the bitstream.

88. 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 1-87.

89. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of claims 1-87.

90. 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:

determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-MBVD mode, the neighboring block being coded with an RRIBC mode; and

generating the bitstream based on the motion information of the current block.

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

determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-MBVD mode, the neighboring block being coded with an RRIBC mode;

generating the bitstream based on the motion information of the current block;

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

92. 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:

determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode, the current block and the neighboring block being coded with an RRIBC mode; and

generating the bitstream based on the motion information of the current block.

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

determining motion information of a current block of the video based on motion information of a neighboring block of the current block and an IBC-TM-MERGE mode, the current block and the neighboring block being coded with an RRIBC mode;

generating the bitstream based on the motion information of the current block; and

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

94. 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:

determining a motion candidate for a current block of the video based on motion information of a neighboring block of the current block and an IBC-AMVP mode, the neighboring block being coded with an RRIBC mode;

determining motion information of the current block based on the motion candidate; and

generating the bitstream based on the motion information of the current block.

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

determining a motion candidate for a current block of the video based on motion information of a neighboring block of the current block and an IBC-AMVP mode, the neighboring block being coded with an RRIBC mode;

determining motion information of the current block based on the motion candidate;

generating the bitstream based on the motion information of the current block; and

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

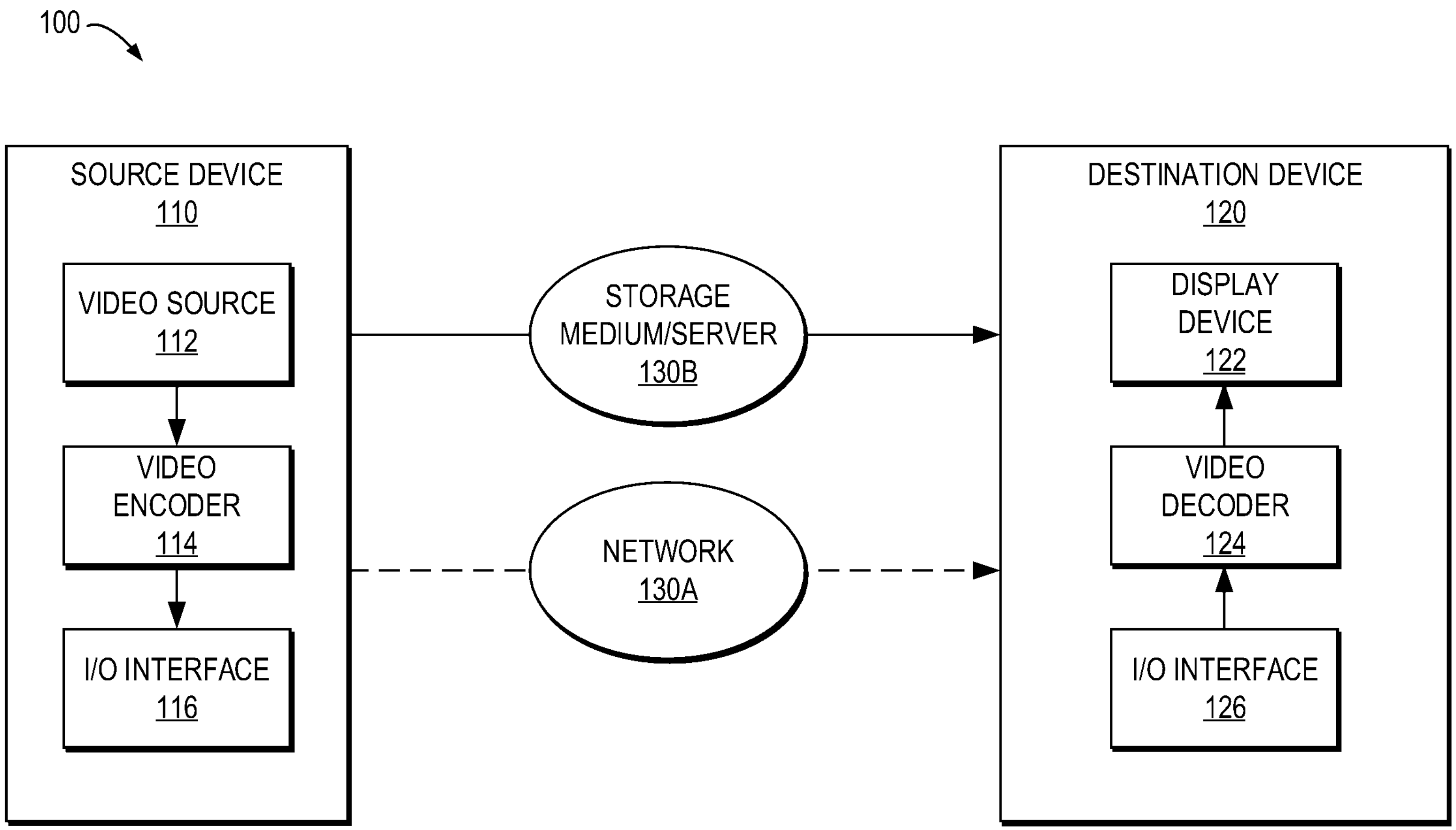
100
SOURCE DEVICE
110
VIDEO SOURCE
112
VIDEO
ENCODER
114
I/O INTERFACE
116
STORAGE
MEDIUM/SERVER
130B
NETWORK
130A
DESTINATION DEVICE
120
DISPLAY
DEVICE
122
VIDEO
DECODER
124
I/O INTERFACE
126

Fig. 1

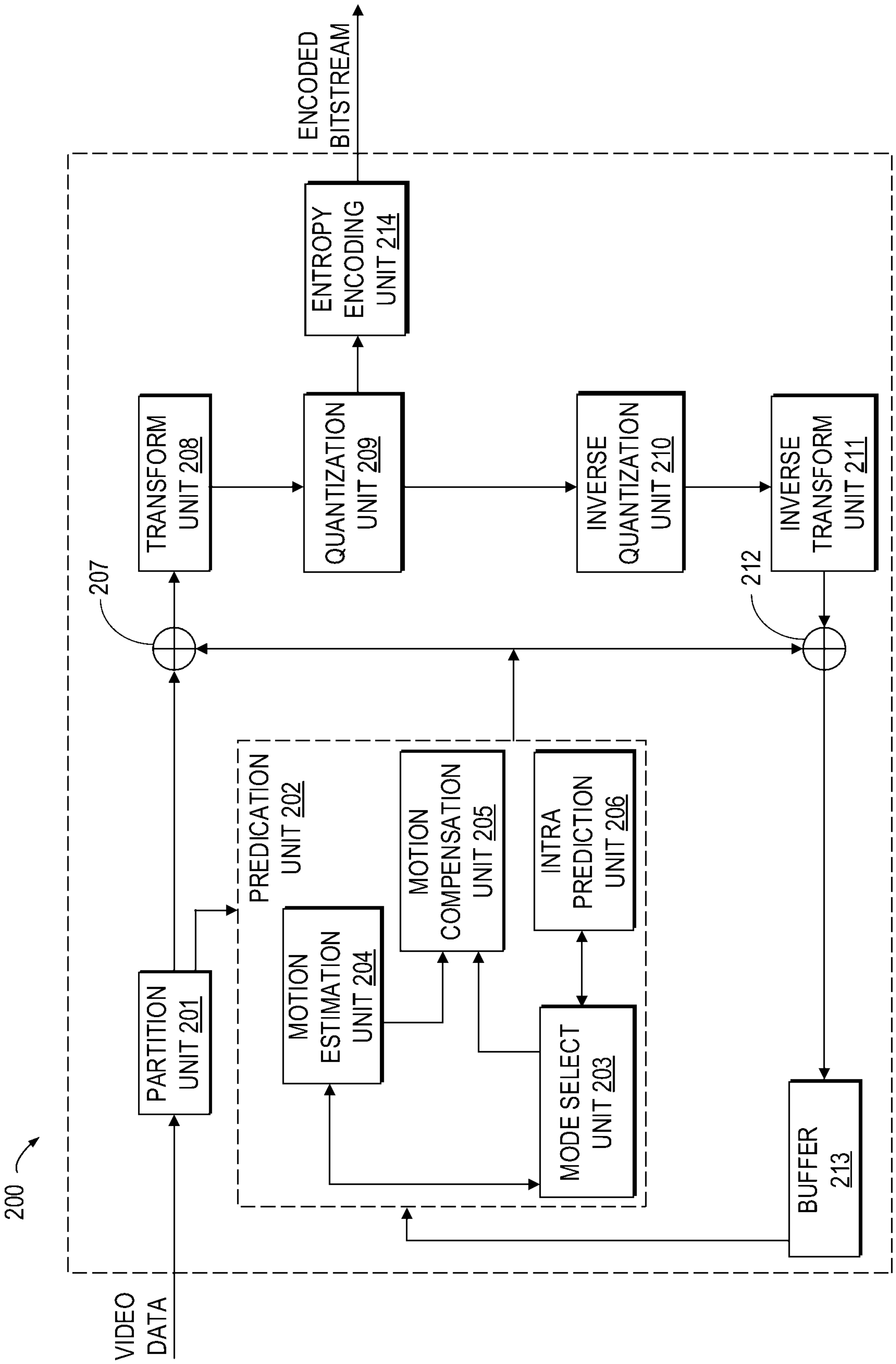
200
VIDEO DATA
PARTITION UNIT 201
PREDICATION UNIT 202
MOTION ESTIMATION UNIT 204
MOTION COMPENSATION UNIT 205
INTRA PREDICTION UNIT 206
MODE SELECT UNIT 203
BUFFER 213
207
TRANSFORM UNIT 208
QUANTIZATION UNIT 209
ENTROPY ENCODING UNIT 214
ENCODED BITSTREAM
INVERSE QUANTIZATION UNIT 210
INVERSE TRANSFORM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

Fig. 2

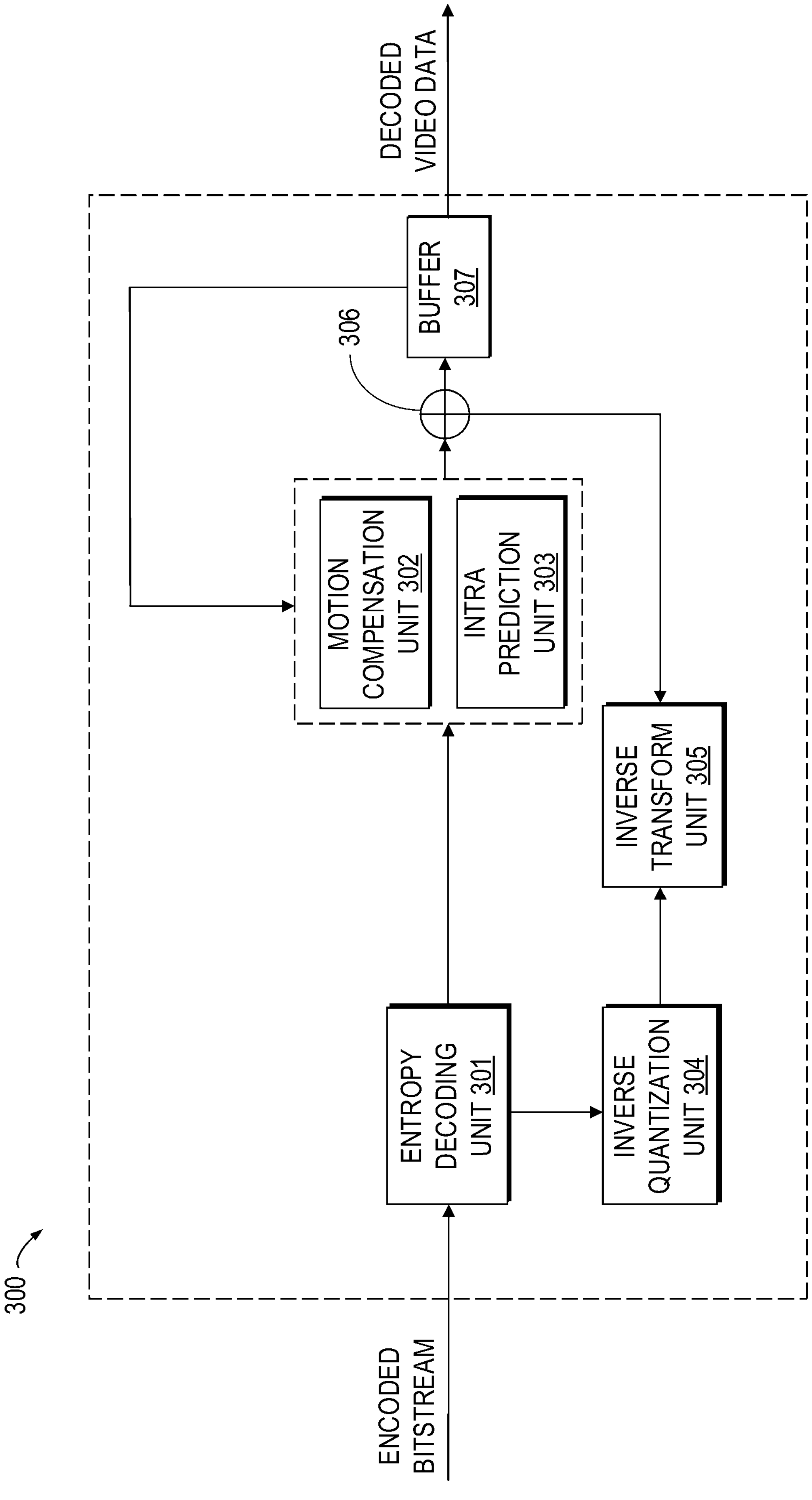
300
ENCODED BITSTREAM
ENTROPY DECODING UNIT 301
INVERSE QUANTIZATION UNIT 304
INVERSE TRANSFORM UNIT 305
MOTION COMPENSATION UNIT 302
INTRA PREDICTION UNIT 303
306
BUFFER 307
DECODED VIDEO DATA

Fig. 3

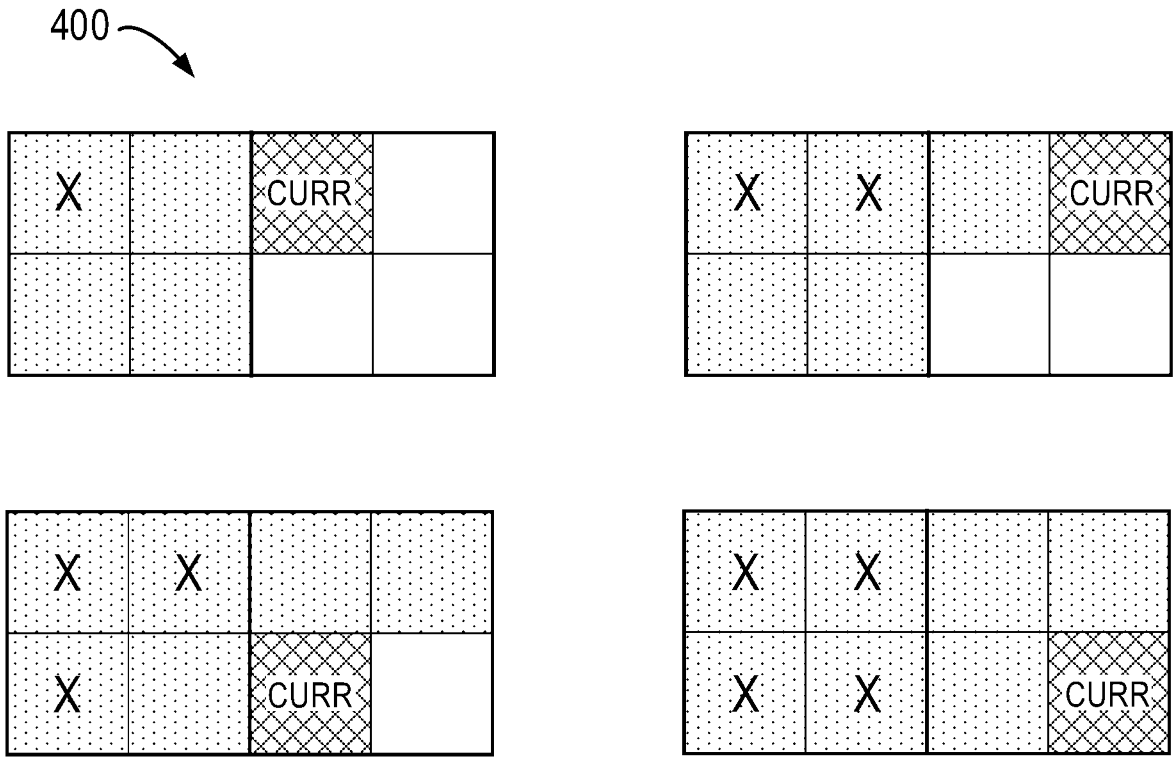

Fig. 4

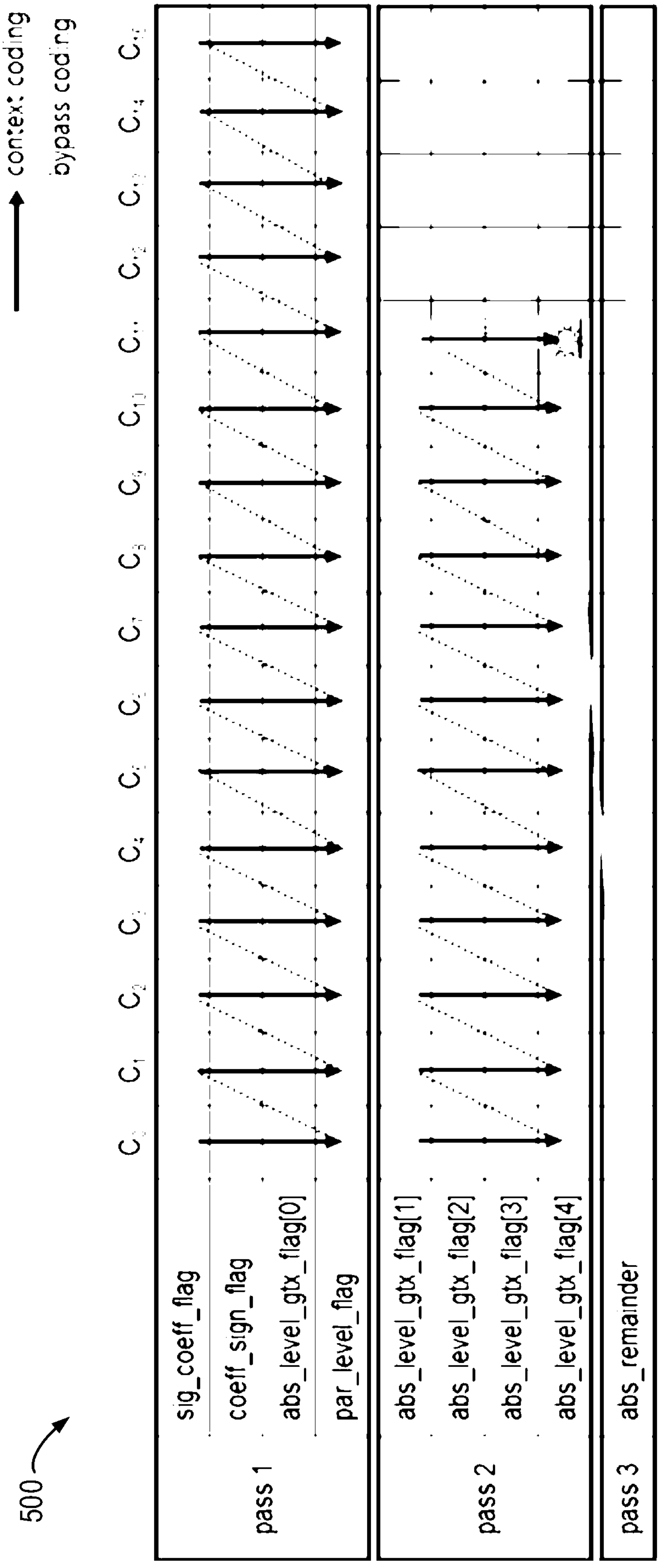
500
context coding
bypass coding
pass 1
sig_coeff_flag
coeff_sign_flag
abs_level_gtx_flag[0]
par_level_flag
pass 2
abs_level_gtx_flag[1]
abs_level_gtx_flag[2]
abs_level_gtx_flag[3]
abs_level_gtx_flag[4]
pass 3
abs_remainder
C0
C1
C2
C3
C4
C5
C6
C7
C8
C9
C10
C11
C12
C13
C14
C15

Fig. 5

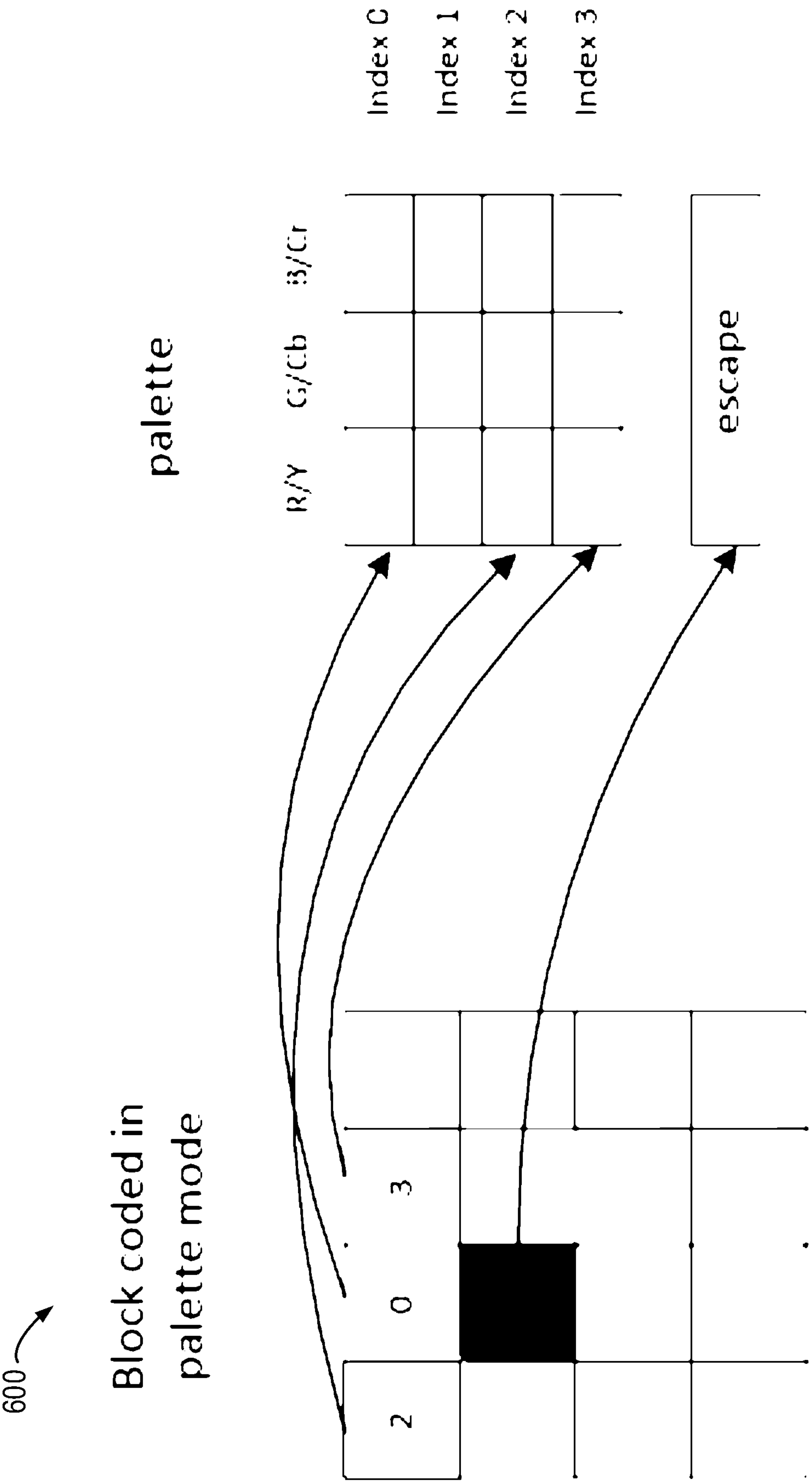
600
Block coded in palette mode
2
0
3
palette
R/Y
G/Cb
B/Cr
Index 0
Index 1
Index 2
Index 3
escape

Fig. 6

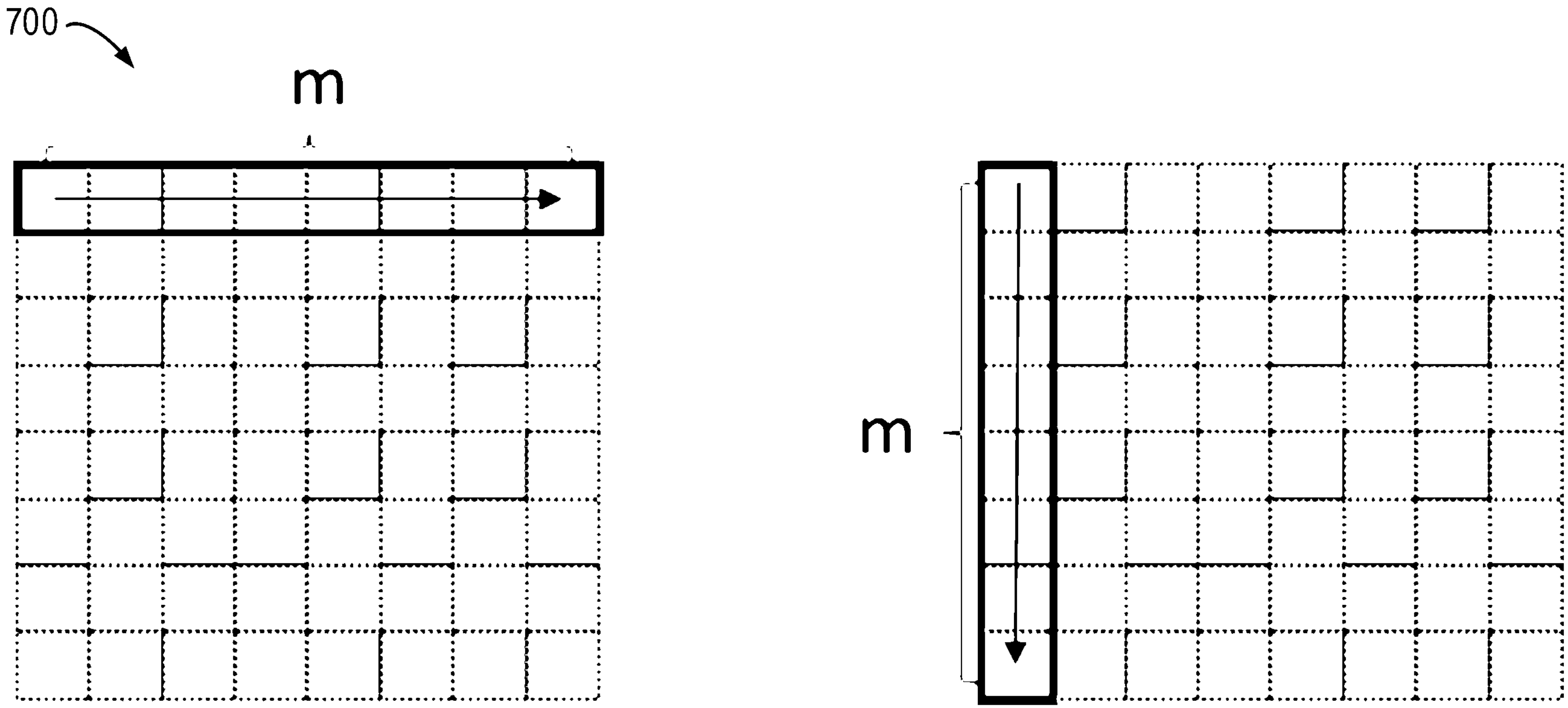

Fig. 7

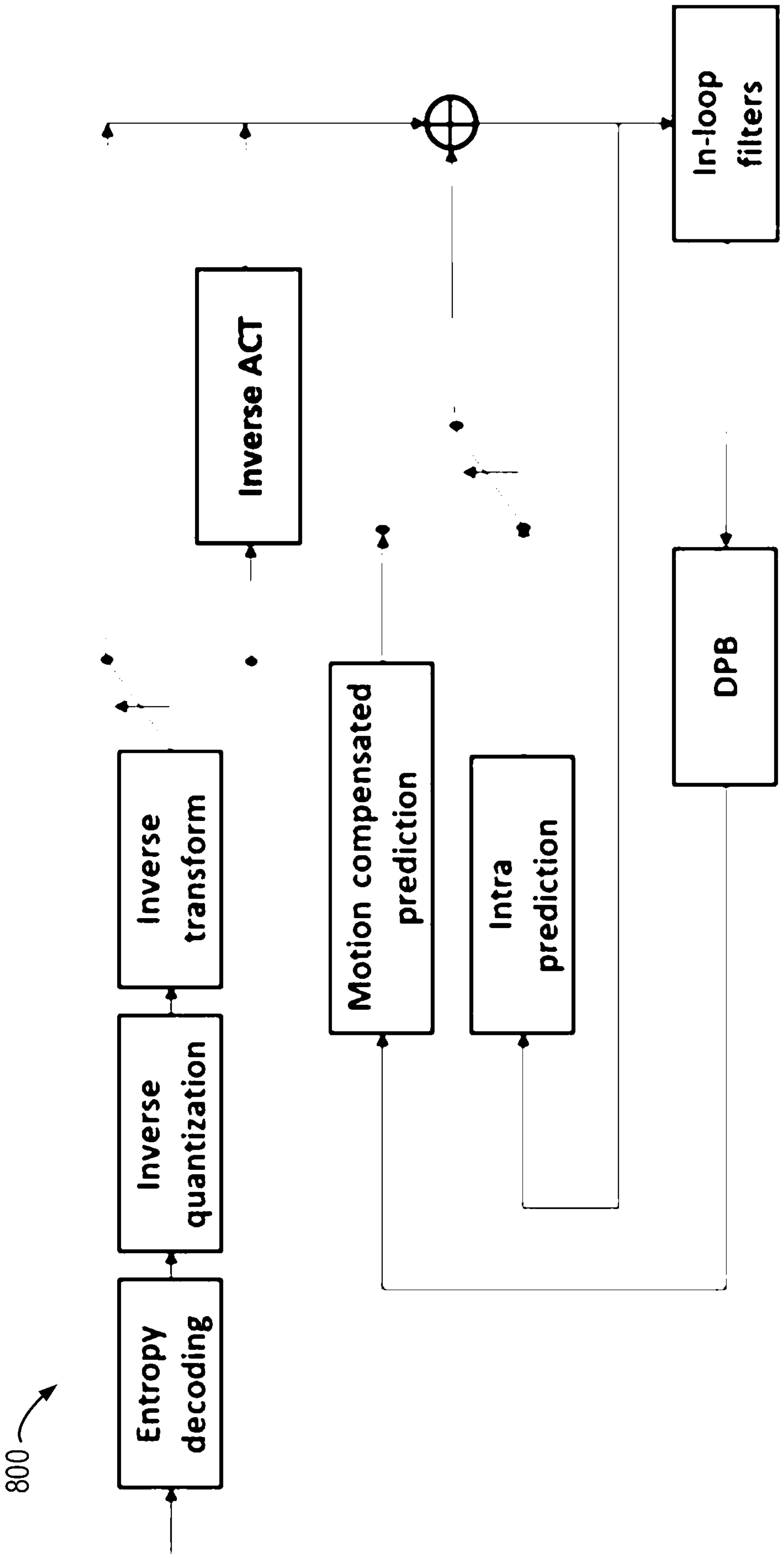
800
Entropy decoding
Inverse quantization
Inverse transform
Inverse ACT
Motion compensated prediction
Intra prediction
DPB
In-loop filters

Fig. 8

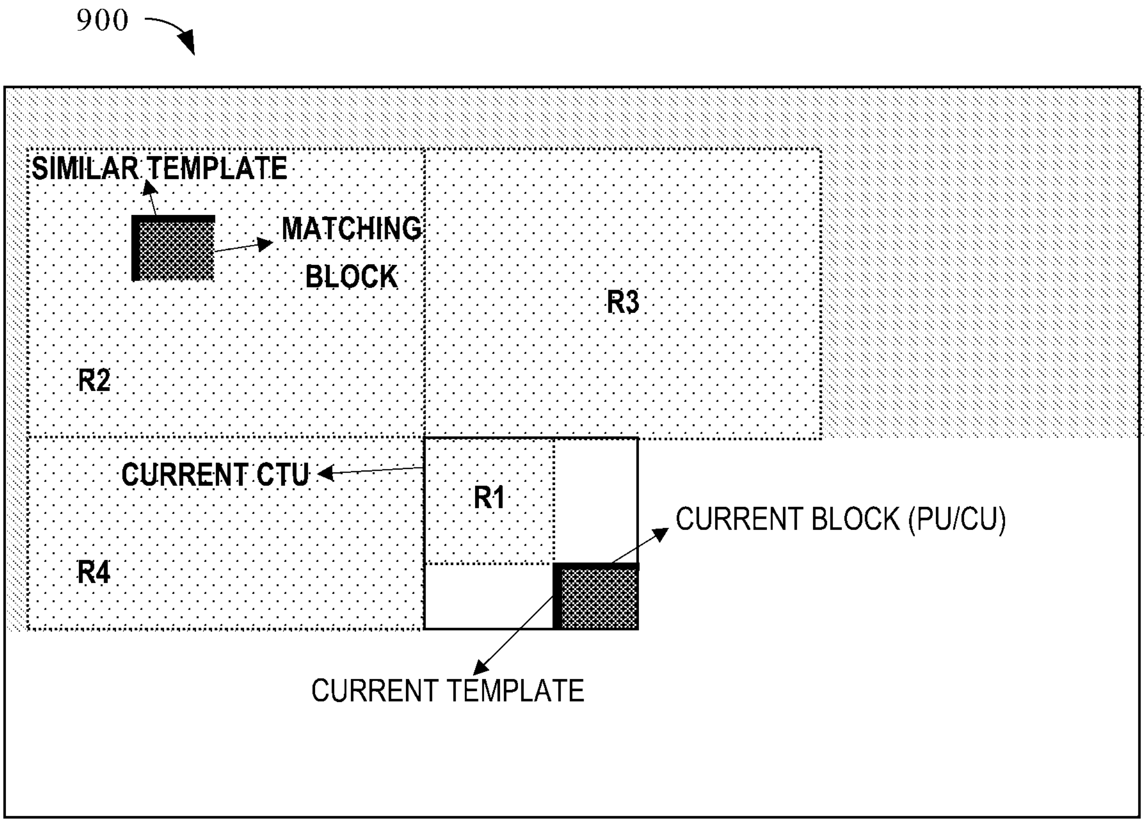
900
SIMILAR TEMPLATE
MATCHING BLOCK
R3
R2
CURRENT CTU
R1
CURRENT BLOCK (PU/CU)
R4
CURRENT TEMPLATE

Fig. 9

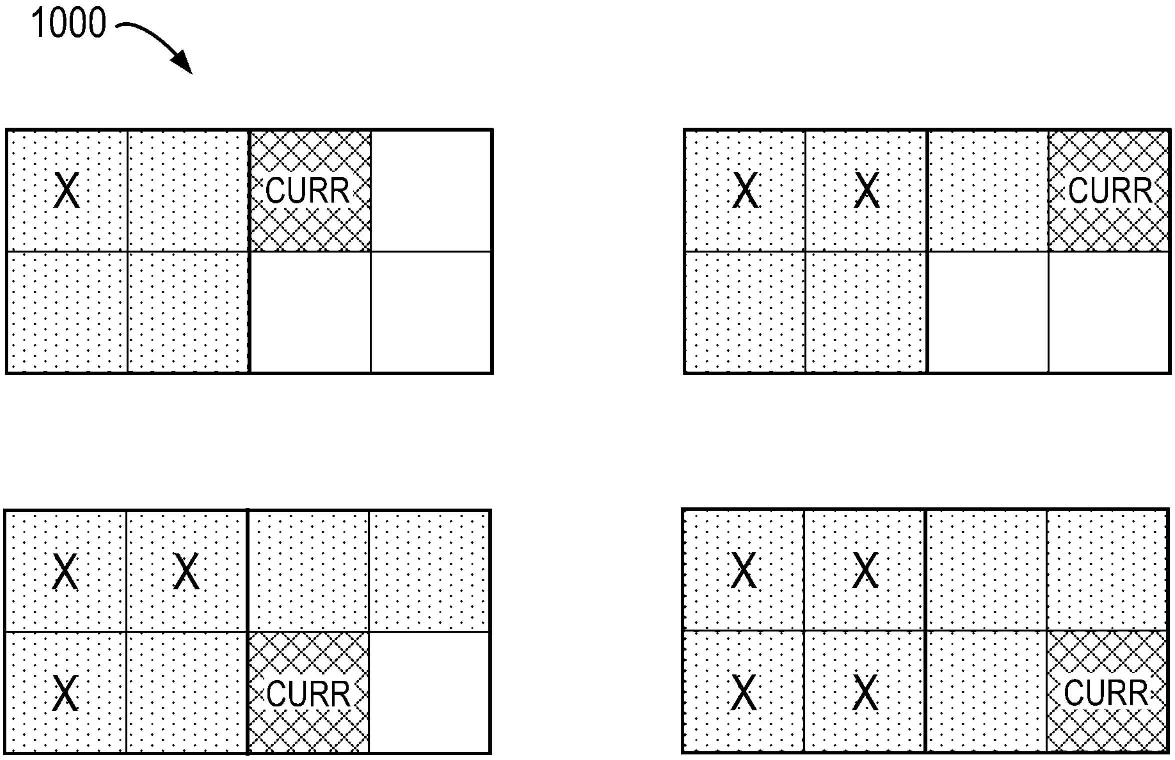

Fig. 10

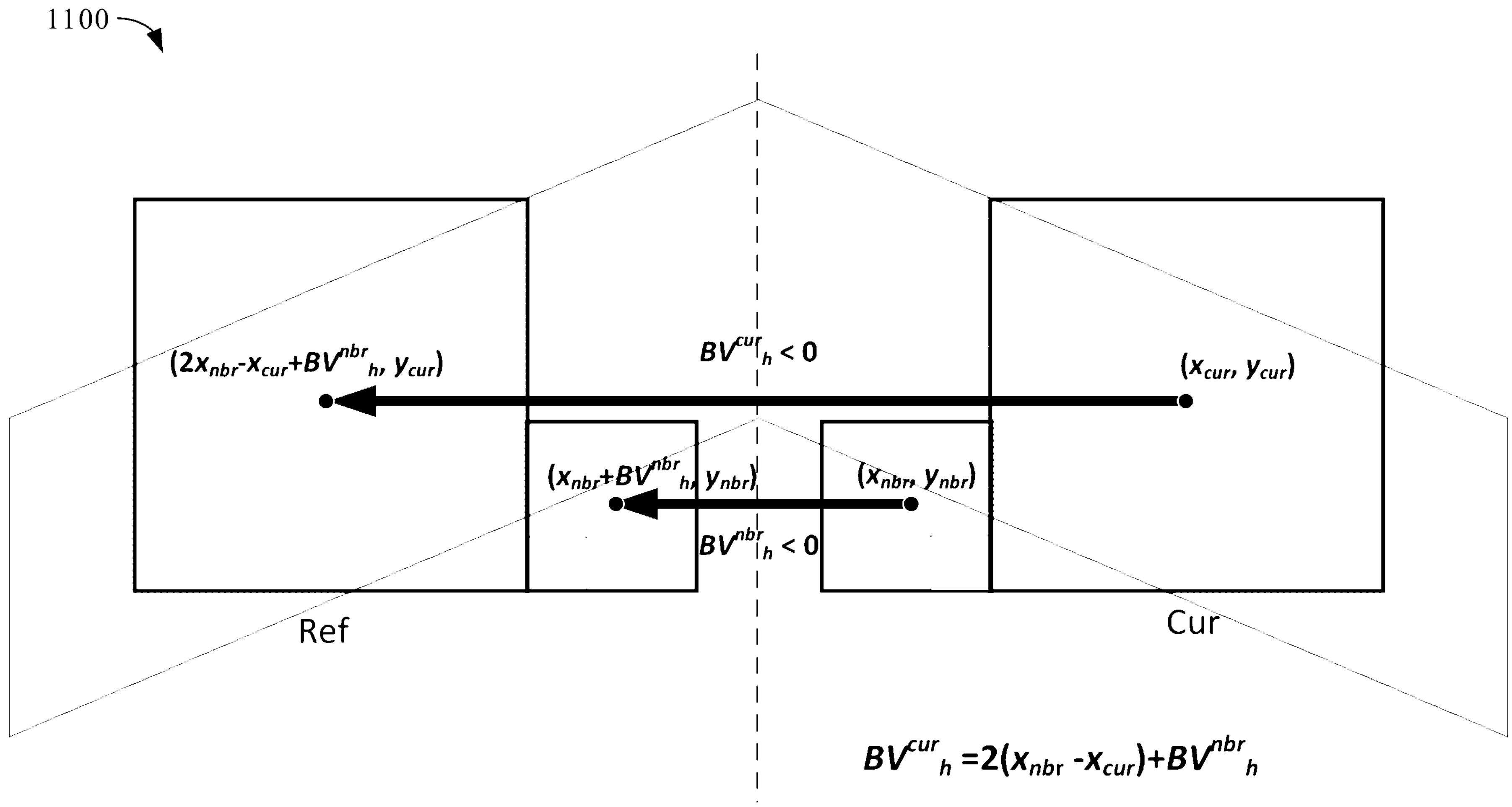
1100
$BV^{cur}_h < 0$
$(2x_{nbr}-x_{cur}+BV^{nbr}_h, y_{cur})$
(x_{cur}, y_{cur})
$(x_{nbr}+BV^{nbr}_h, y_{nbr})$
(x_{nbr}, y_{nbr})
$BV^{nbr}_h < 0$
Ref
Cur
$BV^{cur}_h = 2(x_{nbr} - x_{cur}) + BV^{nbr}_h$

Fig. 11A

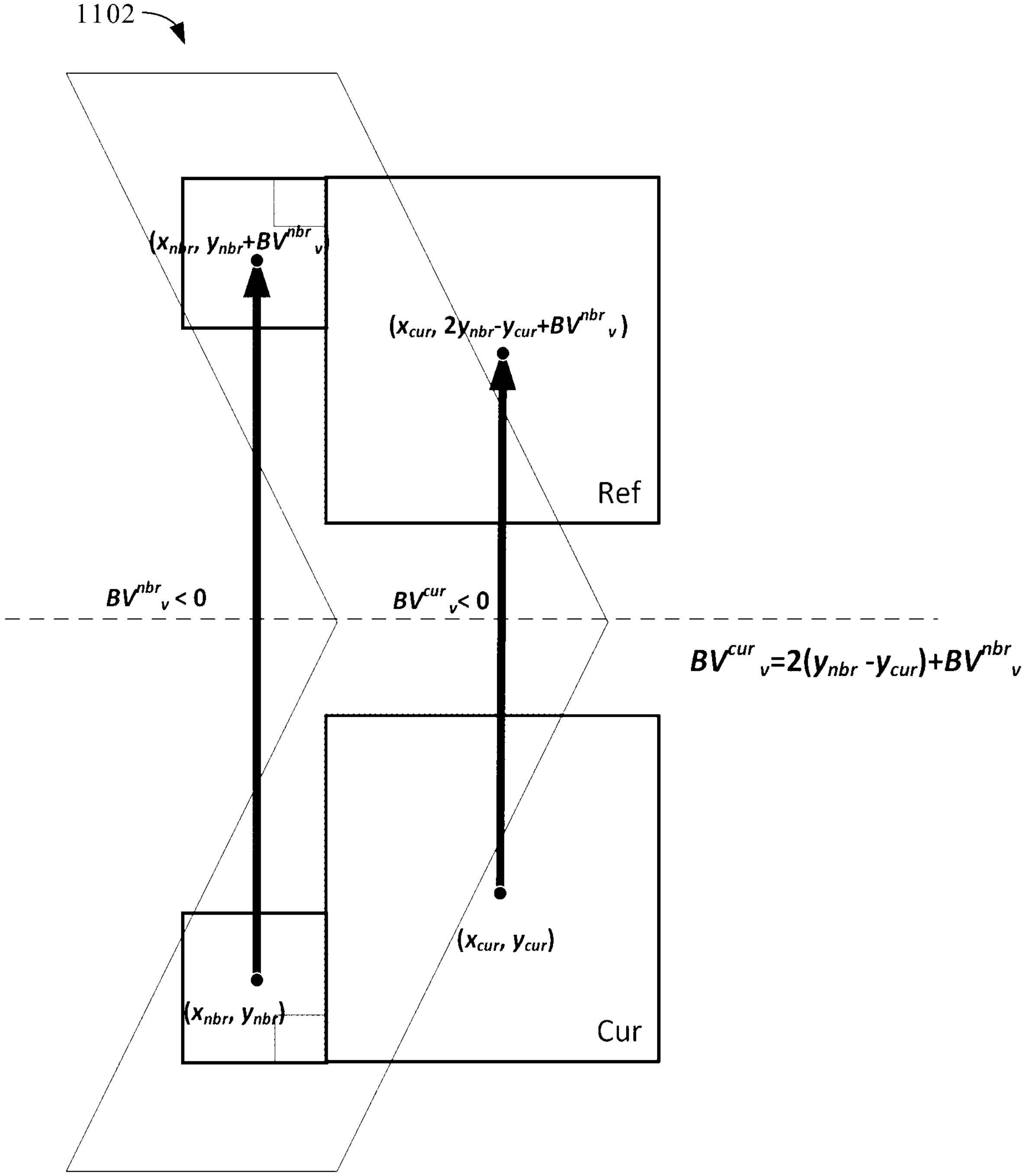
1102
$(x_{nbr}, y_{nbr}+BV^{nbr}_{v})$
$(x_{cur}, 2y_{nbr}-y_{cur}+BV^{nbr}_{v})$
Ref
$BV^{nbr}_{v} < 0$
$BV^{cur}_{v} < 0$
$BV^{cur}_{v}=2(y_{nbr}-y_{cur})+BV^{nbr}_{v}$
(x_{cur}, y_{cur})
(x_{nbr}, y_{nbr})
Cur

Fig. 11B

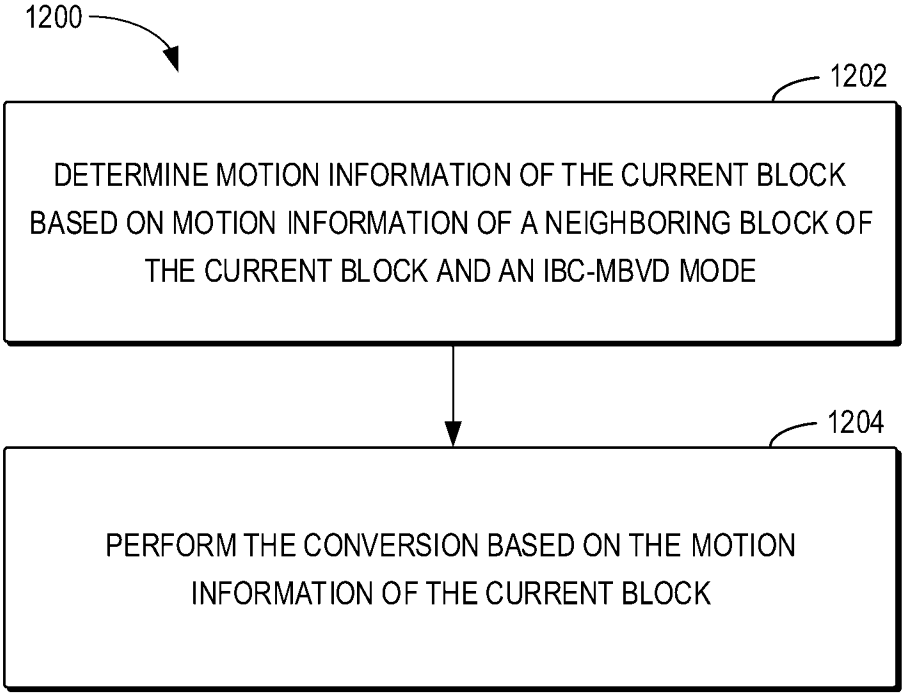
1200
1202
DETERMINE MOTION INFORMATION OF THE CURRENT BLOCK BASED ON MOTION INFORMATION OF A NEIGHBORING BLOCK OF THE CURRENT BLOCK AND AN IBC-MBVD MODE
1204
PERFORM THE CONVERSION BASED ON THE MOTION INFORMATION OF THE CURRENT BLOCK

Fig. 12

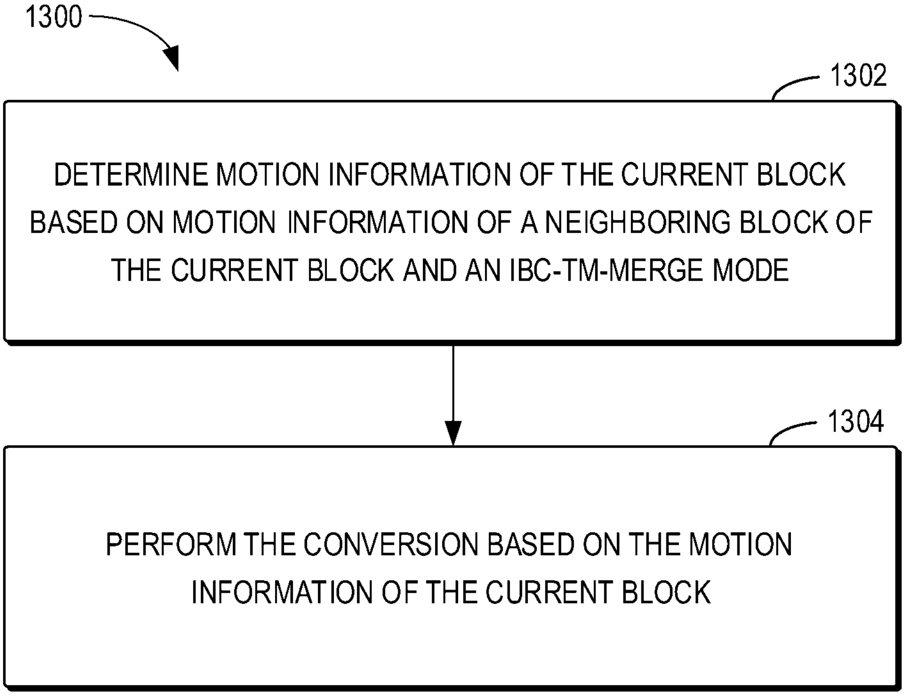
1300
1302
DETERMINE MOTION INFORMATION OF THE CURRENT BLOCK BASED ON MOTION INFORMATION OF A NEIGHBORING BLOCK OF THE CURRENT BLOCK AND AN IBC-TM-MERGE MODE
1304
PERFORM THE CONVERSION BASED ON THE MOTION INFORMATION OF THE CURRENT BLOCK

Fig. 13

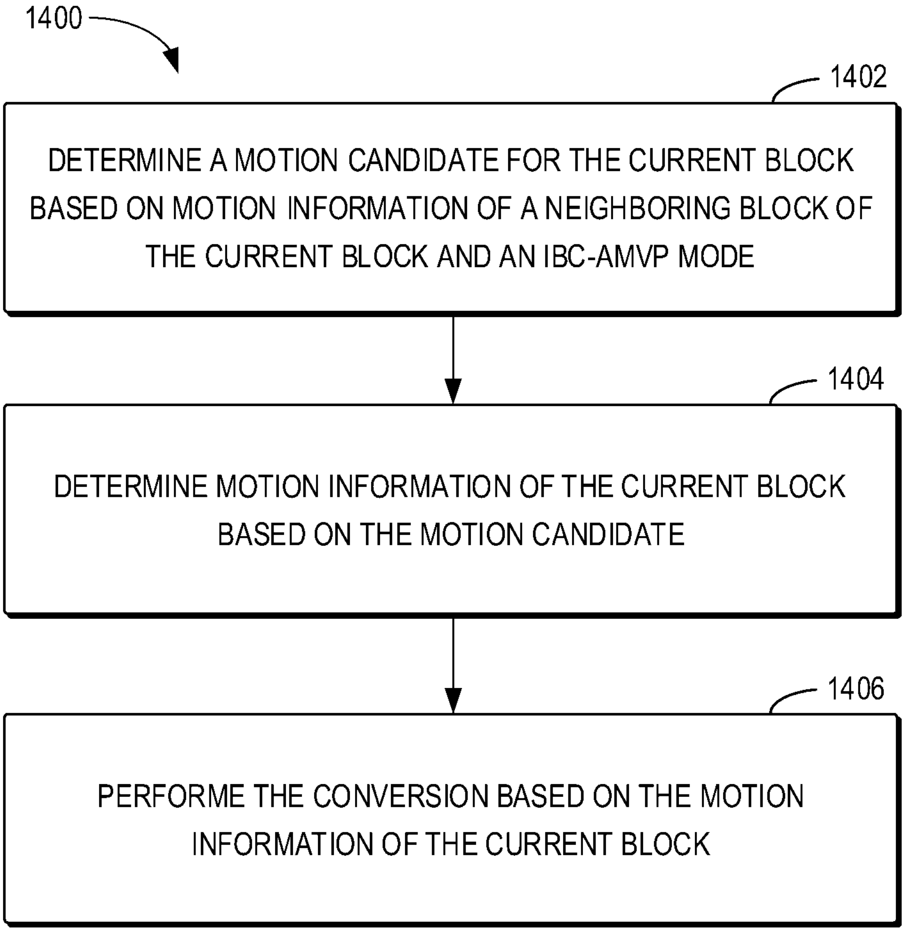
1400
1402
DETERMINE A MOTION CANDIDATE FOR THE CURRENT BLOCK BASED ON MOTION INFORMATION OF A NEIGHBORING BLOCK OF THE CURRENT BLOCK AND AN IBC-AMVP MODE
1404
DETERMINE MOTION INFORMATION OF THE CURRENT BLOCK BASED ON THE MOTION CANDIDATE
1406
PERFORME THE CONVERSION BASED ON THE MOTION INFORMATION OF THE CURRENT BLOCK

Fig. 14

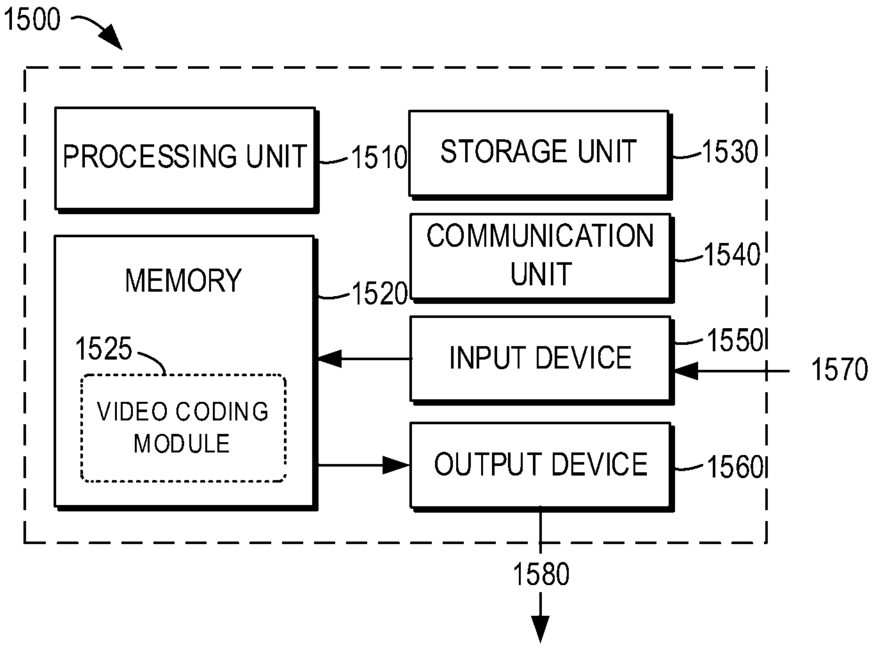
1500
PROCESSING UNIT
1510
STORAGE UNIT
1530
COMMUNICATION UNIT
1540
MEMORY
1520
1525
VIDEO CODING MODULE
INPUT DEVICE
1550
1570
OUTPUT DEVICE
1560
1580

Fig. 15

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2023/105229

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/176(2014.01)i； H04N19/51(2014.01)i； 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,DWPI,ENTXT,ENTXTC,VEN,WPABS,JVET:AMVP,BVD,IBC,MBVD,RRIBC,MVD,TM,merge,motion candidate, reconstruct+,reorder+,neighbo?r,intra block copy ,template match+,merge mode,reconstruction-reordered intra block copy

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	DENG, Zhipin et al. "Non-EE2: Reconstruction-Reordered IBC for screen content coding" *Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29 26th Meeting, by teleconference, 20–29 April 2022,JVET-Z0159-v1,* 29 April 2022 (2022-04-29), Abstract, Section 2	1-41,62-95
Y	CN 110933412 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. et al.) 27 March 2020 (2020-03-27) Claims 1-26	1-41,62-95
A	CN 113302937 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. et al.) 24 August 2021 (2021-08-24) the whole document	1-95
A	CN 113491127 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. et al.) 08 October 2021 (2021-10-08) the whole document	1-95
A	US 2022174303 A1 (LG ELECTRONICS INC.) 02 June 2022 (2022-06-02) the whole document	1-95

☑ 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 **13 September 2023**	Date of mailing of the international search report **21 September 2023**
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 **CHENG,Cong** Telephone No. **(+86) 010-53961725**

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2023/105229

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2020009390 A1 (LG ELECTRONICS INC.) 09 January 2020 (2020-01-09) the whole document	1-95

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2023/105229

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN 110933412 A	27 March 2020	US 2021211654 A1	08 July 2021
		TW 202025730 A	01 July 2020
		WO 2020058893 A1	26 March 2020
		TW 202025731 A	01 July 2020
		WO 2020058894 A1	26 March 2020
		TW 202025780 A	01 July 2020
		WO 2020058895 A1	26 March 2020
		WO 2020058896 A1	26 March 2020
		US 2021211655 A1	08 July 2021
		US 2021211709 A1	08 July 2021
		TW 202025732 A	01 July 2020
		CN 110933411 A	27 March 2020
		CN 110933424 A	27 March 2020
		CN 110933440 A	27 March 2020
CN 113302937 A	24 August 2021	WO 2020147772 A1	23 July 2020
		WO 2020147773 A1	23 July 2020
		US 2021344969 A1	04 November 2021
		US 2021344947 A1	04 November 2021
CN 113491127 A	08 October 2021	WO 2020173482 A1	03 September 2020
		WO 2020173483 A1	03 September 2020
US 2022174303 A1	02 June 2022	KR 20220147731 A	03 November 2022
		WO 2020184991 A1	17 September 2020
		KR 20210118196 A	29 September 2021
WO 2020009390 A1	09 January 2020	None	