



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

(21) International Application Number:
PCT/CN2024/084994

(22) International Filing Date:
29 March 2024 (29.03.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
PCT/CN2023/084816
29 March 2023 (29.03.2023) CN

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

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

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

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

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

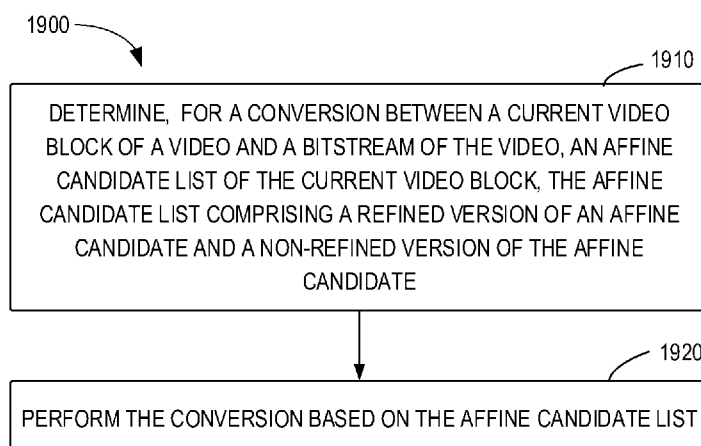


Fig. 19

(57) Abstract: Embodiments of the present disclosure provide a solution for video processing. A method for video processing is proposed. In the method, for a conversion between a current video block of a video and a bitstream of the video, an affine candidate list of the current video block is determined. The affine candidate list comprises a refined version of an affine candidate and a non-refined version of the affine candidate. The conversion is performed based on the affine candidate list.



METHOD, APPARATUS, AND MEDIUM FOR VIDEO PROCESSING

FIELDS

[0001] Embodiments of the present disclosure relates generally to video processing techniques, and more particularly, to affine motion candidate list for video coding.

BACKGROUND

[0002] In nowadays, digital video capabilities are being applied in various aspects of peoples' lives. Multiple types of video compression technologies, such as MPEG-2, MPEG-4, ITU-TH.263, ITU-TH.264/MPEG-4 Part 10 Advanced Video Coding (AVC), ITU-TH.265 high efficiency video coding (HEVC) standard, versatile video coding (VVC) standard, have been proposed for video encoding/decoding. However, coding efficiency of 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 video block of a video and a bitstream of the video, an affine candidate list of the current video block, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; and performing the conversion based on the affine candidate list.

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

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

[0007] In a fourth aspect, another non-transitory computer-readable recording medium

is proposed. The non-transitory computer-readable recording medium stores a bitstream of a video which is generated by a method performed by an apparatus for video processing. The method comprises: determining an affine candidate list of a current video block of the video, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; and generating the bitstream based on the affine candidate list.

[0008] In a fifth aspect, a method for storing a bitstream of a video is proposed. The method comprises: determining an affine candidate list of a current video block of the video, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; generating the bitstream based on the affine candidate list; and storing the bitstream in a non-transitory computer-readable recording medium.

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

BRIEF DESCRIPTION OF THE DRAWINGS

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

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

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

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

[0014] Fig. 4 illustrates positions of spatial and temporal neighboring blocks used in AMVP/merge candidate list construction;

- [0015] Fig. 5 illustrates positions of non-adjacent candidate in ECM;
- [0016] Fig. 6A illustrates a 4-parameter control point based affine motion model;
- [0017] Fig. 6B illustrates a 6-parameter control point based affine motion model;
- [0018] Fig. 7 illustrates an affine MVF per subblock;
- 5 [0019] Fig. 8 illustrates locations of inherited affine motion predictors;
- [0020] Fig. 9 illustrates control point motion vector inheritance;
- [0021] Fig. 10 illustrates locations of candidates position for constructed affine merge mode;
- [0022] Fig. 11 illustrates spatial neighbors for deriving affine merge candidates,
10 where (a) in Fig. 11 is for deriving inherited affine merge candidates, and (B) in Fig. 11 is for deriving constructed affine merge candidates;
- [0023] Fig. 12 illustrates a diagram from non-adjacent neighbors to constructed affine merge candidates;
- [0024] Fig. 13 illustrates an example of generating an HAPC;
- 15 [0025] Fig. 14 illustrates an illustration of regression based affine merge candidate derivation;
- [0026] Fig. 15 illustrates template matching performing on a search area around initial MV;
- [0027] Fig. 16 illustrates a template and the corresponding reference template;
- 20 [0028] Fig. 17 illustrates a template and reference template for block with subblock motion using the motion information of the subblocks of current block;
- [0029] Fig. 18 illustrates a diagram showing deriving sub-CU motion field obtained by applying a motion shift based on the neighboring motion information;
- [0030] Fig. 19 illustrates a flowchart of a method for video processing in accordance
25 with embodiments of the present disclosure; and
- [0031] Fig. 20 illustrates a block diagram of a computing device in which various embodiments of the present disclosure can be implemented.

[0032] Throughout the drawings, the same or similar reference numerals usually refer to the same or similar elements.

DETAILED DESCRIPTION

[0033] 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.

[0034] 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.

[0035] 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.

[0036] 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.

[0037] 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,

5 components and/ or combinations thereof.

Example Environment

[0038] 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
10 110 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
15 input/output (I/O) interface 116.

[0039] 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.

20 [0040] 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,
25 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.

30 [0041] 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
5 be external to the destination device 120 which is configured to interface with an external display device.

[0042] 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
10 standards.

[0043] 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.

[0044] The video encoder 200 may be configured to implement any or all of the
15 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.

[0045] 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
20 213, and an entropy encoding unit 214.

[0046] 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.

[0047] 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
30

example of Fig. 2 separately for purposes of explanation.

[0048] 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.

[0049] 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.

[0050] 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.

[0051] 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.

[0052] 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.

5 [0053] 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
10 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
15 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.

[0054] 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
20 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.

25 [0055] 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.

[0056] In another example, the motion estimation unit 204 may identify, in a syntax
30 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.

[0057] 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.

[0058] 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.

[0059] 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.

[0060] 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.

[0061] 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.

[0062] 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.

[0063] 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.

5 [0064] After the reconstruction unit 212 reconstructs the video block, loop filtering operation may be performed to reduce video blocking artifacts in the video block.

[0065] 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
10 operations to generate entropy encoded data and output a bitstream that includes the entropy encoded data.

[0066] 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.

15 [0067] 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
20 disclosure.

[0068] 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
25 pass generally reciprocal to the encoding pass described with respect to video encoder 200.

[0069] 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
30 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
5 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.

10 **[0070]** 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.

[0071] The motion compensation unit 302 may use the interpolation filters as used by
15 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.

[0072] The motion compensation unit 302 may use at least part of the syntax
20 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
25 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.

[0073] The intra prediction unit 303 may use intra prediction modes for example
30 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.

[0074] 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.

[0075] 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 Affine motion prediction method in video coding. The ideas may be applied individually or in various combination, to any video coding standard or non-standard video codec.

2. Introduction

The exponential increasing of multimedia data poses a critical challenge for video coding. To satisfy the increasing demands for more efficient compression technology, ITU-T and ISO/IEC have developed a series of video coding standards in the past decades. In particular, the ITU-T produced H.261 and H.263, ISO/IEC produced MPEG-1 and MPEG-4 visual, and the two organizations jointly developed the H.262/MPEG-2 Video, H.264/MPEG-4 Advanced Video Coding (AVC), H.265/HEVC and the latest VVC standards. Since H.262/MPEG-2, hybrid video coding framework is employed wherein in intra/inter prediction plus transform

coding are utilized.

2.1.MVP in video coding

Inter prediction aims to remove the temporal redundancy between adjacent frames, which serves as an indispensable component in the hybrid video coding framework. Specifically, inter prediction makes use of the contents specified by motion vector (MV) as the predicted version of the current to-be-coded block, thus only residual signals and motion information are transmitted in the bitstream. To reduce the cost for MV signaling, motion vector prediction (MVP) came into being as an effective mechanism to convey motion information. Early strategies simply use the MV of a specified neighboring block or the median MV of neighboring blocks as MVP. In H.265/HEVC, competing mechanism was involved where the optimal MVP is selected from multiple candidates through rate distortion optimization (RDO). In particular, advanced MVP (AMVP) mode and merge mode are devised with different motion information signaling strategy. With the AMVP mode, a reference index, an MVP candidate index referring to an AMVP candidate list and motion vector difference (MVD) is signaled. Regarding the merge mode, only a merge index referring to a merge candidate list is signaled, and all the motion information associated with the merge candidate is inherited. Both AMVP mode and merge mode need to construct MVP candidate list, and the details of the construction process for these two modes are described as follows.

AMVP mode: AMVP exploits spatial-temporal correlation of motion vector with neighboring blocks, which is used for explicit transmission of motion parameters. For each reference picture list, a motion vector candidate list is constructed by firstly checking availability of left, above temporally neighboring positions, removing redundant candidates and adding zero vector to make the candidate list to be constant length. Fig. 4 illustrates a diagram of positions of spatial and temporal neighboring blocks used in AMVP/merge candidate list construction. For spatial motion vector candidate derivation, two motion vector candidates are eventually derived based on motion vectors of blocks located in five different positions as depicted in Fig. 4. The five neighboring blocks located at B0, B1, B2, and A0, A1 are classified into two groups, where Group A includes the three above spatial neighboring blocks and Group B includes the two left spatial neighboring blocks. The two MV candidates are respectively derived with the first available candidate from Group A and Group B in a predefined order. For temporal motion vector candidate derivation, one motion vector candidate is derived based on two different collocated positions (bottom-right (C0) and central (C1)) checked in order, as depicted in Fig. 4. To avoid redundant MV candidates, duplicated motion vector candidates in the list are abandoned. If the number of potential candidates is smaller than two, additional zero motion vector candidates are added to the list.

Merge mode: Similar to AMVP mode, MVP candidate list for merge mode comprises of spatial and temporal candidates as well. For spatial motion vector candidate derivation, at most four candidates are selected with order A1, B1, B0, A0 and B2 after performing

availability and redundant checking. For temporal merge candidate (TMVP) derivation, at most one candidate is selected from two temporal neighboring blocks (C0 and C1). When there are not enough merge candidates with spatial and temporal candidates, combined bi-predictive merge candidates and zero MV candidates are added to MVP candidate list. Once the number of available merge candidates reaches the signaled maximally allowed number, the merge candidate list construction process is terminated.

In VVC, the construction process for merge mode is further improved by introducing the history-based MVP (HMVP), which incorporates the motion information of previously coded blocks which may be far away from current block. In VVC, HMVP merge candidates are appended to merge list after the spatial MVP and TMVP. In this method, the motion information of a previously coded block is stored in a table and used as MVP for the current CU. The table with multiple HMVP candidates is maintained with first-in-first-out strategy during the encoding/decoding process. Whenever there is a non-subblock inter-coded CU, the associated motion information is added to the last entry of the table as a new HMVP candidate.

During the standardization of VVC, Non-adjacent MVP was proposed to facilitate better motion information derivation by exploiting the non-adjacent area. In ECM software, Non-adjacent MVP are inserted between TMVP and HMVP, where the distances between non-adjacent spatial candidates and current coding block are based on the width and height of current coding block as depicted in Fig. 5, which illustrates a diagram 500 of positions of non-adjacent candidate in ECM.

2.2. Affine motion compensated prediction

In HEVC, only translation motion model is applied for motion compensation prediction (MCP). While in the real world, there are many kinds of motion, e.g., zoom in/out, rotation, perspective motions and the other irregular motions. In VVC, a block-based affine transform motion compensation prediction is applied. Fig. 6A illustrates a diagram 610 of a 4-parameter control point based affine motion model. Fig. 6B illustrates a diagram 620 of a 6-parameter control point based affine motion model. As shown in Fig. 6A and FIG. 6B, the affine motion field of the block is described by motion information of two control point (4-parameter) or three control point motion vectors (6-parameter).

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

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

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

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

Where (mv_{0x}, mv_{0y}) is motion vector of the top-left corner control point, (mv_{1x}, mv_{1y}) is motion vector of the top-right corner control point, and (mv_{2x}, mv_{2y}) is motion vector of the bottom-left corner control point.

- 5 Fig. 7 illustrates an example diagram 700 of affine MVF per subblock. To simplify the motion compensation prediction, block based affine transform prediction is applied. To derive motion vector of each 4×4 luma subblock, the motion vector of the center sample of each subblock, as shown in Fig. 7, is calculated according to above equations, and rounded to 1/16 fraction accuracy. Then the motion compensation interpolation filters are applied to generate the
- 10 prediction of each subblock with derived motion vector. The subblock size of chroma-components is also set to be 4×4. The MV of a 4×4 chroma subblock is calculated as the average of the MVs of the top-left and bottom-right luma subblocks in the collocated 8x8 luma region.

As done for translational motion inter prediction, there are also two affine motion inter

15 prediction modes: affine merge mode and affine AMVP mode.

2.2.1. Affine merge prediction

Affine merge mode can be applied for CUs with both width and height larger than or equal to 8. In this mode the CPMVs of the current CU is generated based on the motion information of the spatial neighboring CUs. There can be up to five CPMVP candidates and an index is

20 signaled to indicate the one to be used for the current CU. In VVC, the following three types of CPVM candidate are used to form the affine merge candidate list:

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

25 MVs of the neighbour CUs,

- Zero MVs.

In VVC, there are maximum two inherited affine candidates, which are derived from affine motion model of the neighboring blocks, one from left neighboring CUs and one from above neighboring CUs. Fig. 8 illustrates an example 800 of locations of inherited affine motion

30 predictors. The candidate blocks are shown in Fig. 8. For the left predictor, the scan order is A0->A1, and for the above predictor, the scan order is B0->B1->B2. Only the first inherited candidate from each side is selected. No pruning check is performed between two inherited candidates. When a neighboring affine CU is identified, its control point motion vectors are used to derive the CPMVP candidate in the affine merge list of the current CU. Fig. 9

35 illustrates an example diagram 900 of control point motion vector inheritance. As shown in

Fig. 9, if the neighbour left bottom block A 910 is coded in affine mode, the motion vectors v_2 , v_3 and v_4 of the top left corner, above right corner and left bottom corner of the CU which contains the block A are attained. When block A is coded with 4-parameter affine model, the two CPMVs of the current CU are calculated according to v_2 , and v_3 . In case that
 5 block A is coded with 6-parameter affine model, the three CPMVs of the current CU are calculated according to v_2 , v_3 and v_4 .

Constructed affine candidate means the candidate is constructed by combining the neighbor translational motion information of each control point. Fig. 10 illustrates a diagram 1000 of locations of Candidates position for constructed affine merge mode. The motion information
 10 for the control points is derived from the specified spatial neighbors and temporal neighbor shown in Fig. 10. CPMV k ($k=1, 2, 3, 4$) represents the k -th control point. For CPMV1, the B2->B3->A2 blocks are checked and the MV of the first available block is used. For CPMV2, the B1->B0 blocks are checked and for CPMV3, the A1->A0 blocks are checked. For TMVP is used as CPMV4 if it's available.

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

{CPMV1, CPMV2, CPMV3}, {CPMV1, CPMV2, CPMV4}, {CPMV1, CPMV3, CPMV4}, {CPMV2, CPMV3, CPMV4}, {CPMV1, CPMV2}, {CPMV1, CPMV3}.

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

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

2.2.2. Affine AMVP prediction

Affine AMVP mode can be applied for CUs with both width and height larger than or equal to 16. An affine flag in CU level is signalled in the bitstream to indicate whether affine AMVP mode is used and then another flag is signalled to indicate whether 4-parameter affine
 30 or 6-parameter affine. In this mode, the difference of the CPMVs of current CU and their predictors CPMVPs is signalled in the bitstream. The affine AMVP candidate list size is 2 and it is generated by using the following four types of CPVM candidate in order:

- Inherited affine AMVP candidates that extrapolated from the CPMVs of the neighbour CUs,
- 35 – Constructed affine AMVP candidates CPMVPs that are derived using the translational MVs of the neighbour CUs,
- Translational MVs from neighboring CUs,
- Zero MVs.

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

- 5 Constructed AMVP candidate is derived from the specified spatial neighbors shown in Fig. 10. The same checking order is used as done in affine merge candidate construction. In addition, reference picture index of the neighboring block is also checked. The first block in the checking order that is inter coded and has the same reference picture as in current CUs is used. There is only one When the current CU is coded with 4-parameter affine mode, and mv_0 and
- 10 mv_1 are both available, they are added as one candidate in the affine AMVP list. When the current CU is coded with 6-parameter affine mode, and all three CPMVs are available, they are added as one candidate in the affine AMVP list. Otherwise, constructed AMVP candidate is set as unavailable. If affine AMVP list candidates is still less than 2 after valid inherited affine AMVP candidates and constructed AMVP candidate are inserted, mv_0 , mv_1 and mv_2 will
- 15 be added, in order, as the translational MVs to predict all control point MVs of the current CU, when available. Finally, zero MVs are used to fill the affine AMVP list if it is still not full.

2.2.3. New Affine candidates derivation methods in ECM-6.0

- In ECM-6.0, 3 additional Affine merge and AMVP candidate derivation methods are integrated, which are Non-adjacent spatial candidates, History-parameter-based candidates
- 20 and Regression based affine candidates.

2.2.3.1. Non-adjacent spatial candidates

- In ECM-6.0, non-adjacent spatial neighbors are investigated to provided candidates for both Affine merge and Affine AMVP. Fig. 11 illustrates of spatial neighbors for deriving affine merge candidates. The pattern of obtaining non-adjacent spatial candidates is shown in Fig. 11.
- 25 Same as the non-adjacent regular merge candidates, the distances between non-adjacent spatial candidates and current coding block are also defined based on the width and height of current CU.

- The motion information of the non-adjacent spatial neighbors in Fig. 11 is utilized to generate additional inherited and constructed affine merge candidates. Specifically, to generate
- 30 inherited candidates, the non-adjacent spatial neighbors are checked based on their distances to the current block, i.e., from near to far. At a specific distance, only the first available neighbor which is coded with Affine mode from each side (e.g., the left and above) of the current block is included. As indicated in (a) of Fig. 11, the checking of the neighbors on the left and above sides are performed from bottom-to-up and right-to-left, respectively. For
- 35 constructed candidates, as shown in the (b) of Fig. 11, the positions of one left and above non-

adjacent spatial neighbors are firstly determined independently; After that, the location of the top-left neighbor can be determined accordingly to form a rectangular virtual block together with the left and above non-adjacent neighbors. Fig. 12 illustrates a diagram 1200 from non-adjacent neighbors to constructed affine merge candidates. The motion information of the three non-adjacent neighbors is used to form the CPMVs at the top-left (A), top-right (B) and bottom-left (C) of the virtual block, which is projected to the current CU to generate the corresponding constructed candidates, as shown in Fig. 12.

2.2.3.2. History-parameter-based affine candidates

History-parameter-based affine model inheritance (HAMI) allows the affine model to be inherited from a previously affine-coded block which may not be neighboring to the current block. A history-parameter table (HPT) is established. An entry of HPT stores a set of affine parameters: a, b, c and d, each of which is represented by a 16-bit signed integer. Entries in HPT is categorized by reference list and reference index. Five reference indices are supported for each reference list in HPT. In a formular way, the category of HPT (denoted as HPTCat) is calculated as

$$\text{HPTCat}(\text{RefList}, \text{RefIdx}) = 5 \times \text{RefList} + \min(\text{RefIdx}, 4) \quad (3)$$

wherein RefList and RefIdx represents a reference picture list (0 or 1) and a reference index, respectively. For each category, at most seven entries can be stored, resulting in 70 entries totally in HPT. At the beginning of each CTU row, the number of entries for each category is initialized as zero. After decoding an affine-coded CU with reference list RefListcur and RefIdxcur, the affine parameters are utilized to update entries in the category HPTCat(RefListcur, RefIdxcur) in a way similar to HMVP table updating.

Fig. 13 illustrates an example diagram 1300 of generating an HAPC. A history-affine-parameter-based candidate (HAPC) is derived from a neighbouring 4×4 block denoted as A0, A1, B0, B1 or B2 in Fig. 13 and a set of affine parameters stored in a corresponding entry in HPT. The MV of a neighbouring 4×4 block served as the base MV. In a formulating way, the MV of the current block at position (x, y) is calculated as:

$$\begin{cases} mv^h(x, y) = a(x - x_{base}) + c(y - y_{base}) + mv_{base}^h \\ mv^v(x, y) = b(x - x_{base}) + d(y - y_{base}) + mv_{base}^v \end{cases}, \quad (4)$$

where (mvhbase, mvvbase) represents the MV of the neighbouring 4×4 block, (xbase, ybase) represents the center position of the neighbouring 4×4 block. (x, y) can be the top-left, top-right and bottom-left corner of the current block to obtain the corner-position MVs (CPMVs) for the current block, or it can be the center of the current block to obtain a regular MV for the current block.

Fig. 13 shows how to derive an HAPC from block A0. The affine parameters {a0, b0, c0, d0} are directly fetched from one entry of category HPTIdx(RefListA0, refIdx0A0) in HPT. The affine parameters from HPT, with the center position of A0 as the base position, and the MV

of block A0 as the base MV, are used together to derive the CPMVs for an affine merge HAPC, or an affine AMVP HAPC. They can also be used to derive MVs located at the center of the current block, as regular merge candidates. A HAPC can be put into the sub-block-based merge candidate list, the affine AMVP candidate list or the regular merge candidate list.

- 5 As a response to new HAPCs being introduced, the size of sub-block-based merge candidate list is increased from five to ten and twelve for random access and low-delay B configurations, respectively. Besides, the size of regular merge candidate list is increased from ten to eleven for random access configurations to accommodate the newly added regular merge candidates.

2.2.3.3. Regression based affine candidate

- 10 In ECM-6.0, the regression based affine merge candidates are derived and added to the affine merge list. Subblock motion field from a previously coded affine CU and motion information from adjacent subblocks of a current CU are used as the input to the regression process to derive proposed affine candidates.

- The previously coded affine CU can be identified from scanning through non-adjacent
15 positions and the affine HMVP table. Fig. 14 illustrates an illustration 1400 of regression based affine merge candidate derivation. Adjacent subblock information of current CU is fetched from 4x4 sub-blocks represented by the grey zone as depicted in Fig. 14. For each sub-block, given a reference list, the corresponding motion vector and center coordinate of the sub-block may be used.

- 20 For each affine CU, up to 2 affine candidates can be derived. One with adjacent subblock information and one without. All the linear-regression-generated candidates are pruned and collected into one candidate sub-group, TM cost based ARMC process is applied when ARMC is enabled. Afterwards, up to N linear-regression-generated candidates are added to the affine merge list when N affine CUs are found.

25 2.3.Template matching merge/AMVP mode in ECM

- Template matching (TM) merge/AMVP mode is a decoder-side MV derivation method to refine the motion information of the current CU by finding the closest match between a template (i.e., top and/or left neighboring blocks of the current CU) in the current picture and a block (i.e., same size to the template) in a reference picture. Fig. 15 illustrates a diagram
30 1500 of template matching performing on a search area around initial MV. As illustrated in Fig. 15, a better MV is to be searched around the initial motion of the current CU within a $[-8, +8]$ -pel search range.

- In AMVP mode, an MVP candidate is determined based on the template matching error to pick up the one which reaches the minimum difference between the current block and the
35 reference block templates, and then TM performs only for this particular MVP candidate for MV refinement. TM refines this MVP candidate, starting from full-pel MVD precision (or 4-

pel for 4-pel AMVR mode) within a $[-8, +8]$ -pel search range by using iterative diamond search. The AMVP candidate may be further refined by using cross search with full-pel MVD precision (or 4-pel for 4-pel AMVR mode), followed sequentially by half-pel and quarter-pel ones depending on AMVR mode. This search process ensures that the MVP candidate still
 5 keeps the same MV precision as indicated by adaptive motion vector resolution (AMVR) mode after TM process.

In the merge mode, similar search method is applied to the merge candidate indicated by the merge index. TM merge may perform all the way down to 1/8-pel MVD precision or skipping those beyond half-pel MVD precision, depending on whether the alternative interpolation
 10 filter (that is used when AMVR is of half-pel mode) is used according to merged motion information. Besides, when TM mode is enabled, template matching may work as an independent process or an extra MV refinement process between block-based and subblock-based bilateral matching (BM) methods, depending on whether BM can be enabled or not according to its enabling condition check. When BM and TM are both enabled for a CU, the
 15 search process of TM stops at half-pel MVD precision and the resulted MVs are further refined by using the same model-based MVD derivation method as in DMVR.

2.4.Adaptive reorder of merge candidates (ARMC)

Inspired by the spatial correlation between reconstructed neighboring pixels and the current coding block, adaptive reorder of merge candidates (ARMC) was proposed to refine the
 20 candidates order in a given candidate list. The underlying assumption is that the candidates with less template matching cost have higher probability to be chosen through RDO process, hence should be placed in front positions within the list to reduce the signaling cost.

The reordering method is applied to regular merge mode, template matching (TM) merge mode, and affine merge mode (excluding the SbTMVP candidate). For the TM merge mode,
 25 merge candidates are reordered before the refinement process.

After a merge candidate list is constructed, merge candidates are divided into several subgroups. The subgroup size is set to 5. Merge candidates in each subgroup are reordered ascendingly according to cost values based on template matching. For simplification, merge candidates in the last but not the first subgroup are not reordered.

The template matching cost is measured by the sum of absolute differences (SAD) between
 30 samples of a template of the current block and their corresponding reference template. Fig. 16 illustrates a diagram 1600 of template and the corresponding reference template. The template comprises a set of reconstructed samples neighboring to the current block, while reference template is located by the same motion information of the current block, as illustrated in Fig.
 35 16. When a merge candidate utilizes bi-directional prediction, the reference samples of the template of the merge candidate are also generated by bi-prediction.

For subblock-based merge candidates with subblock size equal to $W_{sub} * H_{sub}$, the above template comprises several sub-templates with the size of $W_{sub} \times K$, and the left template

comprises several sub-templates with the size of $K \times H_{\text{sub}}$. Fig. 17 illustrates a diagram 1700 of template and reference template for block with sub-block motion using the motion information of the subblocks of current block. As shown in Fig. 17, the motion information of the subblocks in the first row and the first column of current block is used to derive the reference samples of each sub-template.

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

VVC supports the subblock-based temporal motion vector prediction (SbTMVP) method. Similar to the TMVP, SbTMVP takes advantage of the motion field in the collocated picture to facilitate more precise MVP derivation. The same collocated picture used by TMVP is used for SbTMVP. SbTMVP differs from TMVP mainly in two aspects. Firstly, SbTMVP enables sub-CU level motion prediction whereas TMVP predicts motion at CU level; Secondly, compared with TMVP that fetches the temporal MV from the collocated block in the collocated picture (the collocated block is the bottom-right or center block relative to the current CU), SbTMVP applies a motion shift before fetching the temporal motion information from the collocated picture, where the motion shift is obtained by re-using the MV from one of the spatial neighboring blocks of the current CU.

Fig. 18 illustrates the derivation process 1800 of the sub-block level motion field for SbTMVP. In particular, the motion information of left-bottom sub-block A1 is firstly fetched, if either of the MVs in reference list0 and list1 points to the collocated frame, then the corresponding MV will be identified as motion shift. Otherwise, zero mv will be used as motion shift.

Once the motion shift is determined, the specified region in the collocated frame is employed to derive sub-block level motion field. Assuming A1' motion is used as motion shift as depicted in Fig. 18. Then for each sub-CU, the motion information of its corresponding block (the smallest motion grid that covers the center sample) in the collocated picture is fetched to provide motion information, where MV scale operation is firstly performed to align the reference frames of the temporal motion vectors to those of the current CU.

Fig. 18 illustrates deriving sub-CU motion field obtained by applying a motion shift based on the neighboring motion information.

In VVC and ECM, in addition to CU level MVP candidate list, a sub-CU level MVP candidate list is also constructed to provide more precise motion prediction for the current CU, which comprises the motion fields produced by both SbTMVP and AFFINE methods. In particular, only one SbTMVP candidate is included and is always placed in the first entry of the constructed sub-CU level MVP candidate list, whereas multiple AFFINE candidates are included in the list after performing template matching-based reordering, where those with smaller costs are placed in frontier positions.

3. Problems

CPMV is critical for Affine motion compensation since it provides basic motion information for all the sub-blocks within the block. In existing CPMV derivation methods, however, the CPMV of the current block is estimated as the MV of an already-coded block, which may not
 5 guarantee the coherence with the true motion. Therefore, a CPMV refinement method is highly desired to reduce the deviation between the estimated CPMV and the true motion.

4. Detailed solutions

In this disclosure, it is proposed to refine Affine CPMV with template matching. For a given Affine candidate in Affine candidate list, the CPMV may be further refined with template
 10 matching, and the refined Affine candidate is then used to derive sub-block or pixel level Affine motion information for the current block.

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

15 The terms ‘video unit’ or ‘coding unit’ or ‘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.

The terms ‘Affine block’ may represent a block coded with Affine merge, Affine AMVP or any other Affine variant mode (i.e., Affine MMVD etc), which may be described by motion information of two control point (4-parameter) or three control point motion vectors (6-
 20 parameter). The terms ‘CPMV’ may represent the motion information of an Affine block at top-left, top-right and/or bottom-left corners.

The term ‘template’ may represent a reconstructed region that can be used to refine the CPMV, which may represent either ‘separate template’ or ‘unified template’. Here a ‘separate template’ may represent a reconstructed region that can be used to refine individual CPMV,
 25 i.e., specific one(s) of top-left, top-right and/or bottom-left corners, while a ‘unified template’ may represent a reconstructed region that can be used to refine all or arbitrary CPMV(s) for a block. The term ‘template matching cost’ or ‘TM cost’ may represent either matching cost of a separate template or a unified template.

In this disclosure, regarding “a block coded with mode N”, here “mode N” may be a
 30 prediction mode (e.g., MODE_INTRA, MODE_INTER, MODE_PLT, MODE_IBC, and etc.), or a coding technique (e.g., DIMD, TIMD, PDPC, CCLM, CCCM, GLM, intraTMP, AMVP, SMVD, Merge, BDOF, PROF, DMVR, AMVR, TM, Affine, CIIP, GPM, spatial GPM,

SGPM, GPM inter-inter, GPM intra-intra, GPM inter-intra, MHP, GEO, TPM, MMVD, BCW, HMVP, SbTMVP, LIC, OBMC, ALF, deblocking, SAO, bilateral filter, LMCS, and the corresponding variants, and etc.).

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. In one example, affine motion compensation may be refined by using previously decoded samples.

a) In one example, at least one CPMV may be refined.

10 b) In one example, at least one MV of one subblock of affine motion compensation may be refined.

c) In one example, at least one affine parameter (such as a , b , c , d , e , f) may be refined.

d) In one example, previously decoded samples may be a template of the current block.

15 e) In one example, previously decoded samples may be a template of the reference block.

f) In one example, template represents a reconstructed region that can be used to refine the CPMV.

g) In one example, for an Affine-coded block, different separate templates may be used for different control points.

20 i. In one example, for a control point, the corresponding separate template may comprise the samples from the adjacent and/or non-adjacent positions in the already-reconstructed region.

1) In one example, the separate templates for all the control points are collected from adjacent reconstructed region.

25 2) In one example, the separate template samples for some control points are collected from the adjacent reconstructed region of the current block, while for the rest control points, the template samples are collected from non-adjacent reconstructed region.

30 a) In one example, specifically, the template samples for left-top corner are collected from non-adjacent region, while for right-top and/or left-bottom corners, the template samples are collected from adjacent region.

3) In one example, both adjacent and non-adjacent samples are used for certain control points.

- ii. In one example, for different control points, the shape of the separate templates may be different.
 - 1) In one example, for certain control points, L-shape (e.g., comprising both above and left neighbouring samples) separate template is used.
 - 2) In one example, for certain control points, I-shape or ‘—’ shape template (e.g., comprising either left or above (but not both) neighbouring samples) may be used.
- iii. In one example, which shape of template is used for a CPMV refinement may be based on the position/location of the control point.
 - 1) In one example, CPMV at the top-left corner of the current video unit may use a L-shape template (e.g., comprising both above and left neighbouring samples).
 - 2) In one example, CPMV at the top-right corner of the current video unit may use a ‘—’ shape template (e.g., comprising above neighbouring samples only).
 - 3) In one example, CPMV at the top-left corner of the current video unit may use a I-shape template (e.g., comprising left neighbouring samples only).
 - 4) In one example, for a certain CPMV, the shapes of templates in the current picture and reference picture are same.
 - a) For example, as depicted in Fig. 16, the template of a CPMV may refer to a group of neighbouring samples in the current picture (e.g., templates in the current picture) and a second group of neighbouring samples in a reference picture (e.g., templates in the reference picture).
- iv. In one example, for different control points, the number of the samples used in the template may be different.
 - 1) Alternatively, the number of the sample used for different control points are the same for an Affine block.
 - 2) For different control points, the lines (or rows or columns) of the samples used in the template may be different.
- h) In one example, a unified template is used during the CPMV refinement.
 - i. In one example, the TM cost associated with the unified template is used to determine the MV shift value.

- ii. In one example, the TM cost associated with the unified template is used to determine the CPMV combination.
 - iii. In one example, a unified template may comprise all or partial adjacent samples of the whole block, i.e. as shown in Fig. 16.
- 5 i) A template may only comprise samples from one component such as luma, or from multiple components such as luma and chroma.
- j) In one example, for arbitrary template, a reference template region with the same shape may be located with a MV, as shown in Fig. 16.
- 10 k) In one example, a template may not necessarily contain all the pixels in a certain region, it may contain part of the pixels in the specified region.
2. When constructing an Affine candidate list, CPMV refinement may be firstly performed to a potential affine candidate, then the refined candidate is inserted into the Affine candidate list.
- 15 a) In one example, alternatively, CPMV refinement is conducted after the Affine candidate list is constructed.
- i. In one example, only the Affine candidate(s) with specific index(es) need to perform CPMV refinement.
 - ii. In one example, all or partial Affine candidates need to perform CPMV refinement.
 - 20 iii. In one example, similarity check is firstly conducted to determine whether a candidate needs to perform CPMV refinement or not.
 - 1) In one example, the candidates in the list are traversed in a certain order. The i -th candidate C_i may only not need to perform refinement when it is similar enough compared with arbitrary candidate in $\{C_1, \dots, C_{i-1}\}$.
- 25 a) In one example, the traverse order may be derived based on TM cost.
- b) In one example, specifically, a pair of candidates may be considered similar enough if the all the CPMV difference are smaller than a threshold, and/or the same prediction direction, and or reference frame is used.
- 30 c) In one example, the candidates in $\{C_1, \dots, C_{i-1}\}$ may already be refined by TM.
3. In one example, a first Affine candidate list is constructed firstly, followed by a second Affine candidate list construction process.
- a) For example, the input of the second Affine candidate list generation may be based on the output of the first Affine candidate list generation.
- 35 b) For example, the first Affine candidate list may be constructed without CPMV refinement.
- c) For example, the second Affine candidate list may be generated by applying CPMV refinement on the CPMV candidates in the first Affine candidate list.

- i. For example, at least one CPMV candidate in the first Affine candidate list may be refined.
- ii. Alternatively, more than one CPMV candidates in the Affine candidate list may be refined.
- 5 iii. For example, the CPMV refinement may be based on TM.
- d) For example, the first Affine candidate list may be constructed with a candidate reordering process.
 - i. For example, the reordering process may be based on TM.
- e) For example, the second Affine candidate list may be constructed without any candidate reordering process.
- 10 f) For example, different pruning rules may be used in the first pruning and the second pruning.
 - i. For example, the first Affine candidate list generation may be applied associated with a first pruning method.
 - 15 ii. For example, the second Affine candidate list generation may be applied associated with a second pruning method.
 - iii. For example, the threshold for motion similarity checks in the first and second pruning methods may be different.
 - iv. For example, a block dimension (e.g., block width and/or height) based threshold
 - 20 v. For example, alternatively, a fixed threshold may be used in the second pruning method.
- 4. For a given Affine candidate, some or all of the CPMVs may be refined based on TM, then the refined CPMVs are used to derive the Affine motion information for the current
- 25 block and/or sub-block(s).
- a) In one example, both integer and fractional precision may be used to refine the control points.
 - i. In one example, only integer precision is used to refine the control point, and fractional precision searching is skipped.
 - 30 1) In one example, whether fractional precision search is needed depends on the results of integer precision search.
 - ii. In one example, it is proposed to use a specific interpolation filter to generate reference templates for motion vectors pointing to fractional positions.
 - 1) In one example, a simplified interpolation filter may be applied.

- 2) In one example, the simplified interpolation filter can be 2-tap bilinear, alternatively, it can also be 4-tap, 6-tap or 8-tap filter that belongs to DCT, DST, Lanczos or any other interpolation types.
 - 3) In one example, a more complex interpolation filter (e.g., with longer filter taps) may be applied.
- iii. In one example, whether to use above methods (e.g., integer precision, different interpolation filters) or not and/or how to use above methods can be signalled in the bitstream (such as in SPS, PPS, picture header, slice header, CTU, CU, etc.) or determined on-the-fly according to decoded information.
- 1) In one example, which method to be applied may be dependent on the coding tool.
 - 2) In one example, which method to be applied may be dependent on block dimension.
- b) In one example, different control points are respectively refined, which means the MV shift values (i.e., the difference between an initial CPMV and the corresponding refined CPMV) may be different for different control points.
- i. In one example, all or some of the control points may first be respectively refined by TM, then one combination of control points is determined by looping over all or some of the combinations (i.e. M (such as M = 4) combinations for 4-parameter model, - N (such as N = 8) combinations for 6-parameter model) of CPMVs before and after refinement, and one set of CPMVs that minimize the TM cost of the current block is derived.
- 1) In one example, in above case, all or some of the control points may first be refined by the respective separate templates.
 - 2) In one example, for each combination of CPMVs, the sub-block level motion information is calculated for the boundary sub-blocks, then the unified TM cost is calculated in accordance with the method described in section 2.4 and Fig. 17. The optimal combination which yields the least TM cost is selected as the refined Affine candidate.
 - a) In one example, only partial boundary sub-blocks need to calculate the TM cost.
 - 3) In one example, alternatively, there is no need to loop over all the combinations, and the combination of which the control points are all refined by TM is directly served as the refined Affine candidate.

4) In one example, when the refined Affine candidate is derived, a 2nd pass control point refinement may be performed to further refine each control point.

a) In one example, each CPMV is further iteratively refined to minimize the TM cost of the current block. In each iteration, one CPMV is refined while the others are fixed.

c) In one example, alternatively, multiple control points are simultaneously refined, where a same MV shift value is shared for all or multiple control points.

i. In one example, all or partial MV shift values in a given MV shift set are traversed one by one. The MV shift value being traversed is assigned to all or multiple CPMVs, then the motion information of the boundary sub-blocks associated with the refined CPMVs are calculated, and the TM cost is formulated accordingly. In this process, the one that yields the least TM cost is determined as the best motion shift value, which may be finally used to refine the CPMVs.

1) In one example, specifically, multiple integer MV shift values are respectively traversed, the one yielding the least TM cost are determined as the initial search point for fractional shift value.

d) In one example, the refined Affine candidate may replace the original one.

i. In one example, the refined Affine candidate will always replace the original one.

ii. In one example, alternatively, the refined Affine candidate will conditionally replace the original one.

1) In one example, specifically, the TM costs associated with the original CPMVs (termed as $C_{beforeTM}$) and the refined CPMVs ($C_{afterTM}$) are respectively calculated, and the refined Affine candidate will replace the original one only when the ratio of $C_{afterTM}$ and $C_{beforeTM}$ is smaller (or larger) than a constant or an adaptively determined value TH .

a) In one example, in above case, different coding modes, e.g., Affine merge/Affine AMVP/Affine MMVD, may have different TH value settings.

iii. Alternatively, the refined Affine candidate may be used as a new candidate.

1) In one example, the refined Affine candidate may be placed in a position adjacent to (i.e., right before or after) the original one in the Affine candidate list.

- 2) In one example, alternatively, the refined Affine candidates may be placed in arbitrary positions in the Affine candidate list.
 - 3) In one example, a refined affine candidate may be compared with at least one candidate already in the candidate list. If they are the same or similar, then it is not added into the list.
- 5
5. The CPMV refinement may be used with regression based Affine candidate derivation method.
 - a) In one example, after all or some of the CPMVs are refined with TM (yielding *Affine_model_TM*), the motion information of the boundary sub-blocks associated with *Affine_model_TM* are derived, which is then fed to a regression model to output a new Affine model (termed as *Affine_model_R*). Then the TM cost of boundary sub-blocks with *Affine_model_TM* and *Affine_model_R* are respectively calculated and compared. And the one with less TM cost is determined as the ultimate refined Affine candidate.
- 10
- i. In one example, all or some of the CPMVs may firstly perform integer precision TM refinement (yielding *Affine_model_TM_I*), then perform fractional precision TM refinement (yielding *Affine_model_TM_F*). And the motion information of the boundary sub-blocks associated with *Affine_model_TM_I* is derived, which is then fed to a regression model to output a new Affine model (*Affine_model_R*). Finally, the TM cost of boundary sub-blocks with *Affine_model_TM_F* and *Affine_model_R* are calculated and compared, and the one with less TM cost is determined as the ultimate refined Affine candidate.
 - ii. In one example, only partial sub-blocks may need to calculate TM cost to generate *Affine_model_TM*, *Affine_model_TM_I* and/or *Affine_model_TM_F*.
- 15
- 20
6. In one example, TM-based refinement may be applied to affine merge or affine AMVP (affine inter).
 - a) In one example, the MVP(s) of affine AMVP may be refined based on TM.
 - i. Alternatively, the MVP(s) of affine AMVP may be refined based on DMVR.
 - b) In one example, for affine AMVP (inter) mode, whether TM-based refinement is performed or not may be dependent on the precision of MV or MVD.
 - i. In one example, specifically, TM-based refinement is performed only when specific MV/MVD precision is used for a block.
 - c) In one example, different MV shift sets or searching procedures may be used for affine merge and affine AMVP (affine inter).
 - i. In one example, specifically, different number of MV shift value may be used for affine merge and affine AMVP (affine inter).
- 25
- 30
- 35

- ii. In one example, for affine AMVP (affine inter), there is no need to compute TM cost for certain MV shift if the initial CPMVs and that with the MV shift yields the same CPMVs after rounding to certain precision.
- 7. In one example, TM-based refinement may be applied to an affine-coded block together with DMVR-based refinement.
- a) In one example, TM-based refinement may be applied before DMVR.
- b) In one example, TM-based refinement may be applied after DMVR.
- c) Alternatively, TM-based refinement may be applied to an affine-coded block exclusively with DMVR-based refinement.
- 8. In one example, derivation of TM cost may depend on whether the block is bi-predicted or uni-predicted.
- a) If the block is bi-predicted, TM cost may be derived based on bi-prediction on TM.
 - i. In one example, let TM_{ref0} and TM_{ref1} be the reference TM associated with the List0 and List1 respectively, then the ultimate reference TM (TM_{bi}) may be derived as:

$$TM_{bi} = a * TM_{ref0} + (1 - a) * TM_{ref1}.$$

- 1) In one example, a equals to 0.5.
- 2) In one example, a is determined based on BCW index.
- 3) In one example, TM_{ref0} is generated based on the CPMVs in List 0, and/or TM_{ref1} is generated based on the CPMVs in List 1.
- b) Alternatively, if the block is bi-predicted, TM cost may be calculated for List0 and List1 separately.
- 9. In one example, the refinement of CPMVs may be done in an iterative way.
 - a) For example, in one step of refinement, one CPMV is refined while others are fixed.
 - b) In one example, the already refined CPMV(s) may be used when a subsequent CPMV is to be refined.
 - i. In one example, alternatively, the CPMV before refinement is used when the subsequent CPMVs are to be refined.
 - c) In one example, the refinement of CPMVs may be done in an iterative way for bi-predicted blocks.
 - i. In one example, the CPMVs associated with List K ($K=0$ or 1) may be firstly refined, then the CPMVs associated with List $(1-K)$ are refined.
 - 1) Whether to and/or how to refine the CPMVs in a later List $(1-K)$ may be determined based on the refined CPMVs of a former List K.

- ii. In one example, the CPMVs associated with List 0 and List 1 may be separately refined.
 - 1) In one example, specifically, when the CPMVs in List K ($K=0$ or 1) are being refined, for each searching step, uni-directional reference TM in List K is generated based on the corresponding CPMVs, and TM cost is hereby calculated to determine the best MV shift value.
 - iii. In one example, alternatively, the CPMVs associated with List 0 and List 1 may be jointly refined.
 - 1) In one example, specifically, when the CPMVs in List K ($K=0$ or 1) are being refined, for each searching step, bi-directional reference TM is generated based on the CPMV information of both List (as described in Bullet 8). The one yielding the least TM cost is determine as the best MV shift value.
10. Multiple rounds of refinement may be conducted to CPMVs.
- a) In one example, all or partial CPMVs may be refined in each round of refinement.
 - b) In one example, all or partial CPMVs may already be refined in a former round refinement, then a later round is conducted to further refine the CPMVs.
11. Whether to and/or how to refine the CPMVs based on TM may be determined based on the prediction direction of the current block.
- a) In one example, the CPMVs may need to be refined by TM only when the current block is uni-predicted.
 - b) In one example, the CPMVs may need to be refined by TM only when the current block is bi-predicted.
 - c) In one example, the CPMVs may always need to be refined by TM no matter whether the current block is bi-predicted or not.
12. More than one CPMV refinement processes may be cascaded for an affine coded block.
- a) For example, an affine DMVR process may be applied based on refined CPMVs.
 - i. For example, a first step of CPMV refinement for affine mode (e.g., affine merge, and/or affine amvp) may be an explicit (MMVD based) and/or implicit (e.g., TM based) CPMV refinement process.
 - 1) For example, for the explicit refinement, block level syntax elements such as MMVD index, and/or MMVD step, and/or MMVD distance, may be signalled in the bitstream.
 - 2) For example, for the implicit refinement, the CPMVs may be refined at the decoder side (e.g., TM based) without block level signalling.
 - ii. For example, a second step of CPMV refinement for affine mode (e.g., affine merge, and/or affine amvp) may be a DMVR based refinement process (e.g., regression based affine DMVR, and/or affine DMVR which adding translation off-

sets to CPMVs), and the affine merge DMVR process may be conducted based on the refined CPMVs obtained at the first step.

1) Furthermore, for example, two different affine DMVR processed may be applied, e.g., one is regression based and the other is not regression based.

5 iii. For example, the bilateral cost may be computed and used to determine the DMVR based offsets.

b) Alternatively, the CPMVs may be firstly refined based on DMVR, then afterwards, the DMVR refined CPMVs may be further refined by an explicit (MMVD based) and/or implicit (e.g., TM based) CPMV refinement process.

10 13. For example, pixel/sample based affine prediction may be applied based on the refined CPMVs.

a) For example, the refined CPMVs may be derived based on explicit method (e.g., MMVD based), and/or TM based refinement, and/or DMVR based refinement.

15 b) For example, the refined CPMVs may be applied for uni-directional affine prediction, and/or bi-directional affine prediction.

c) For example, the refined CPMV based pixel/sample affine prediction may be applied for affine merge, and/or affine AMVP mode.

14. The disclosed methods may be applied to MHP (Multiple hypothesis prediction) coded block if Affine prediction is used as a hypothesis.

20 15. The Affine candidate list may simultaneously contain two alternative versions (e.g., refined and non-refined version) of an Affine candidate.

a) In one example, the original candidate and a TM-refined (or bilateral matching-refined) version of the same candidate may simultaneously appear in the Affine list.

25 i. In one example, whether two alternative versions appear in the Affine list for a same candidate or not may depend on the prediction direction (e.g., bi- or uni-predicted) of the candidate.

ii. In one example, after a preliminary Affine list is constructed, the candidates in the list may be checked in some certain order (e.g., TM cost). If the current candidate satisfies certain condition (e.g., uni-predicted), then the alternative version of the candidate is used to append the list.

30 1) In one example, if the current candidate is already refined, then the alternative version (i.e., non-refined version) of the candidate may be used to append the list, and vice versa.

35 2) In one example, the alternative version may be used to replace certain or arbitrary candidate.

a) In one example, the alternative version may be used to replace the candidate with or after a specific index A .

i. In one example, A may be a constant value.

40 ii. In one example, the index of the first candidate with all-zero CPMVs is determined as A .

iii. In one example, A may be dependent on the candidate number of certain type(s) of Affine candidate.

iii. In one example, the ultimate Affine candidate index is determined based on the initial Affine index I and a variable B .

45 1) In one example, the initial Affine index I may be parsed in the bitstream.

- 2) In one example, B may be arbitrary value ranging from 0 to the maximum allowed Affine candidate number.
- 3) In one example, the index of the first candidate with all-zero CPMVs in the Affine list may be determined as B .
- 5 4) In one example, if I is smaller than B , then I is used to specify the Affine candidate.
 - a) Alternatively, if I is larger than or equal to B , then an adjusted index is derived, which is used to specify Affine candidate.
 - i. In one example, the adjusted index may be $(I - B)$.
 - 10 ii. In one example, the adjusted index may be dependent on I , B , and/or the prediction direction of the candidates.
 - iii. In one example, the alternative version of the candidate specified by the adjusted index may be used as the ultimate Affine candidate.
 - 15 1. In one example, If the candidate specified by the adjusted index is already refined, then the non-refined version will be used to generate Affine prediction, and vice versa.
16. Whether to and/or how to apply the disclosed methods above may be determined based on syntax element(s).
 - a) For example, at least one syntax element is signalled in the bitstream.
 - 20 b) For example, whether to and/or how to apply the disclosed methods 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.
 - c) For example, whether to and/or how to apply the disclosed methods may be signalled
 - 25 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.
 - d) For example, 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.
 - 30 e) For example, whether a syntax element (i.e., indicating if TM refinement is applied to CPMVs) is signalled or not may be determined based on another syntax element.

[0076] Further details will be described below. Fig. 19 illustrates a flowchart of a method 1900 for video processing in accordance with embodiments of the present disclosure. The method 1900 may be implemented for a conversion between a current video block of a video and a bitstream of the video.

[0077] At block 1910, an affine candidate list of the current video block is determined. The affine candidate list comprises a refined version of an affine candidate and a non-

refined version of the affine candidate.

[0078] At block 1920, the conversion is performed based on the affine candidate list. In some embodiments, the conversion includes encoding the current video block into the bitstream. Alternatively, or in addition, in some embodiments, the conversion includes
5 decoding the current video block from the bitstream.

[0079] The method 1900 enables applying an affine candidate list with two versions of an affine candidate. In this way, the coding effectiveness and coding efficiency can be improved.

[0080] In some embodiments, the refined version of the affine candidate comprises a
10 template matching refined affine candidate or a bilateral matching refined affine candidate.

[0081] In some embodiments, whether the refined and non-refined versions of the affine candidate are in the affine candidate list is based on a prediction direction of the affine candidate, the prediction direction comprising a bi-predicted direction or a uni-
15 predicted direction.

[0082] In some embodiments, a plurality of affine candidates in a preliminary affine candidate list is checked based on an order, and if a current affine candidate in a first version satisfies a condition, a second version of the current affine candidate is used to append the affine candidate list.

20 [0083] In some embodiments, the order is based on a template matching cost.

[0084] In some embodiments, the condition indicates that the current affine candidate is coded with a uni-predicted direction.

[0085] In some embodiments, the current affine candidate is refined, and the second version of the current affine candidate comprises a non-refined version of the current
25 affine candidate. Alternatively, in some embodiments, the current affine candidate is non-refined, and the second version of the current affine candidate comprises a refined version of the current affine candidate.

[0086] In some embodiments, a plurality of affine candidates in a preliminary affine candidate list is checked based on an order, and if a current affine candidate in a first
30 version satisfies a condition, a second version of the current affine candidate is used to

replace a target affine candidate in the affine candidate list.

[0087] In some embodiments, the target affine candidate is with a first index or after the first index. For example, the first index may be a constant value.

[0088] In some embodiments, an index of a first candidate with all-zero control point motion vectors is determined as the first index.

[0089] In some embodiments, the first index is based on the number of candidates of a type of the affine candidate.

[0090] In some embodiments, an ultimate index of an affine candidate in the affine candidate list is based on an initial index of the affine candidate and a variable.

10 **[0091]** In some embodiments, the initial index is parsed in the bitstream.

[0092] In some embodiments, the variable comprises a value in a range from 0 to a maximum number of allowed affine candidates in the affine candidate list.

[0093] In some embodiments, an index of a first candidate with all-zero control point motion vectors in the affine candidate list is determined as the variable.

15 **[0094]** In some embodiments, if the initial index of the affine candidate is smaller than the variable, the ultimate index of the affine candidate is the initial index.

[0095] In some embodiments, if the initial index of the affine candidate is larger than or equal to the variable, the ultimate index of the affine candidate is determined by adjusting the initial index at least in part based on the variable.

20 **[0096]** In some embodiments, the ultimate index is determined by subtracting the variable from the initial index.

[0097] In some embodiments, the ultimate index is determined based on at least one of: the initial index, the variable, or a prediction direction of the affine candidate.

[0098] In some embodiments, a version of the affine candidate specified by the
25 ultimate index is used as an ultimate affine candidate.

[0099] In some embodiments, if the version of the affine candidate specified by the ultimate index is refined, a non-refined version of the affine candidate is used to generate an affine prediction.

[00100] In some embodiments, if the version of the affine candidate specified by the ultimate index is non-refined, a refined version of the affine candidate is used to generate an affine prediction.

5 [00101] In some embodiments, whether to and/or how to apply the method 1900 is based on a syntax element in the bitstream.

[00102] In some embodiments, the syntax element is at at least one of: a sequence level, a group of pictures level, a picture level, a slice level, or a tile group level.

10 [00103] In some embodiments, the syntax element is included in at least one of: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a decoded parameter set (DPS), decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter set (APS), a slice header, or a time group header.

[00104] In some embodiments, the syntax element is indicated in a region containing more than one sample or pixel.

15 [00105] In some embodiments, the region comprises one of: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, or a subpicture.

20 [00106] In some embodiments, whether to and/or how to apply the method 1900 is determined based on coding information of the current video block.

25 [00107] In some embodiments, the coding information comprises at least one of: a block size of the current video block, a color format of the current video block, a single or dual tree partitioning of the current video block, a color component of the current video block, a slice type of the current video block, or a picture type of the current video block.

[00108] In some embodiments, whether a first syntax element indicating if a template matching based refinement process is applied to a control point motion vector of the current video block is determined based on a second syntax element.

30 [00109] 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, an affine candidate list of a current video block of the video is determined. The affine candidate list comprises a refined version of an affine candidate and a non-refined version of the affine candidate.

5 The bitstream is generated based on the affine candidate list.

[00110] According to still further embodiments of the present disclosure, a method for storing bitstream of a video is provided. In the method, an affine candidate list of a current video block of the video is determined. The affine candidate list comprises a refined version of an affine candidate and a non-refined version of the affine candidate.

10 The bitstream is generated based on the affine candidate list. The bitstream is stored in a non-transitory computer-readable recording medium.

[00111] 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.

[00112] Clause 1. A method for video processing, comprising: determining, for a
15 conversion between a current video block of a video and a bitstream of the video, an affine candidate list of the current video block, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; and performing the conversion based on the affine candidate list.

[00113] Clause 2. The method of clause 1, wherein the refined version of the affine
20 candidate comprises a template matching refined affine candidate or a bilateral matching refined affine candidate.

[00114] Clause 3. The method of clause 1 or 2, wherein whether the refined and non-
refined versions of the affine candidate are in the affine candidate list is based on a prediction direction of the affine candidate, the prediction direction comprising a bi-
25 predicted direction or a uni-predicted direction.

[00115] Clause 4. The method of any of clauses 1-3, wherein a plurality of affine candidates in a preliminary affine candidate list is checked based on an order, and if a current affine candidate in a first version satisfies a condition, a second version of the current affine candidate is used to append the affine candidate list.

30 [00116] Clause 5. The method of clause 4, wherein the order is based on a template matching cost.

[00117] Clause 6. The method of clause 4, wherein the condition indicates that the current affine candidate is coded with a uni-predicted direction.

[00118] Clause 7. The method of clause 4, wherein the current affine candidate is refined, and the second version of the current affine candidate comprises a non-refined version of the current affine candidate, or wherein the current affine candidate is non-refined, and the second version of the current affine candidate comprises a refined version of the current affine candidate.

[00119] Clause 8. The method of any of clauses 1-3, wherein a plurality of affine candidates in a preliminary affine candidate list is checked based on an order, and if a current affine candidate in a first version satisfies a condition, a second version of the current affine candidate is used to replace a target affine candidate in the affine candidate list.

[00120] Clause 9. The method of clause 8, wherein the target affine candidate is with a first index or after the first index.

[00121] Clause 10. The method of clause 9, wherein the first index is a constant value.

[00122] Clause 11. The method of clause 9, wherein an index of a first candidate with all-zero control point motion vectors is determined as the first index.

[00123] Clause 12. The method of clause 9, wherein the first index is based on the number of candidates of a type of the affine candidate.

[00124] Clause 13. The method of any of clauses 1-12, wherein an ultimate index of an affine candidate in the affine candidate list is based on an initial index of the affine candidate and a variable.

[00125] Clause 14. The method of clause 13, wherein the initial index is parsed in the bitstream.

[00126] Clause 15. The method of clause 13 or 14, wherein the variable comprises a value in a range from 0 to a maximum number of allowed affine candidates in the affine candidate list.

[00127] Clause 16. The method of clause 13, wherein an index of a first candidate with all-zero control point motion vectors in the affine candidate list is determined as the variable.

[00128] Clause 17. The method of any of clauses 13-16, wherein if the initial index of the affine candidate is smaller than the variable, the ultimate index of the affine candidate is the initial index.

[00129] Clause 18. The method of any of clauses 13-17, wherein if the initial index of the affine candidate is larger than or equal to the variable, the ultimate index of the affine candidate is determined by adjusting the initial index at least in part based on the variable.

[00130] Clause 19. The method of clause 18, wherein the ultimate index is determined by subtracting the variable from the initial index.

[00131] Clause 20. The method of clause 18, wherein the ultimate index is determined based on at least one of: the initial index, the variable, or a prediction direction of the affine candidate.

[00132] Clause 21. The method of any of clauses 13-20, wherein a version of the affine candidate specified by the ultimate index is used as an ultimate affine candidate.

[00133] Clause 22. The method of clause 21, wherein if the version of the affine candidate specified by the ultimate index is refined, a non-refined version of the affine candidate is used to generate an affine prediction.

[00134] Clause 23. The method of clause 21, wherein if the version of the affine candidate specified by the ultimate index is non-refined, a refined version of the affine candidate is used to generate an affine prediction.

[00135] Clause 24. The method of any of clauses 1-23, wherein whether to and/or how to apply the method is based on a syntax element in the bitstream.

[00136] Clause 25. The method of clause 24, wherein the syntax element is at least one of: a sequence level, a group of pictures level, a picture level, a slice level, or a tile group level.

[00137] Clause 26. The method of clause 24 or clause 25, wherein the syntax element is included in at least one of: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a decoded parameter set (DPS), decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter set (APS), a slice header, or a time group header.

[00138] Clause 27. The method of any of clauses 24-26, wherein the syntax element is indicated in a region containing more than one sample or pixel.

[00139] Clause 28. The method of clause 27, wherein the region comprises one of: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, or a subpicture.

[00140] Clause 29. The method of any of clauses 1-28, wherein whether to and/or how to apply the method is determined based on coding information of the current video block.

10 [00141] Clause 30. The method of clause 29, wherein the coding information comprises at least one of: a block size of the current video block, a color format of the current video block, a single or dual tree partitioning of the current video block, a color component of the current video block, a slice type of the current video block, or a picture type of the current video block.

15 [00142] Clause 31. The method of any of clauses 1-30, wherein whether a first syntax element indicating if a template matching based refinement process is applied to a control point motion vector of the current video block is determined based on a second syntax element.

20 [00143] Clause 32. The method of any of clauses 1-31, wherein the conversion includes encoding the current video block into the bitstream.

[00144] Clause 33. The method of any of clauses 1-31, wherein the conversion includes decoding the current video block from the bitstream.

25 [00145] Clause 34. 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 1-33.

[00146] Clause 35. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of clauses 1-33.

30 [00147] Clause 36. 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 an affine candidate list of a current video block of the video, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; and generating
5 the bitstream based on the affine candidate list.

[00148] Clause 37. A method for storing a bitstream of a video, comprising: determining an affine candidate list of a current video block of the video, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; generating the bitstream based on the affine candidate
10 list; and storing the bitstream in a non-transitory computer-readable recording medium.

Example Device

[00149] Fig. 20 illustrates a block diagram of a computing device 2000 in which various embodiments of the present disclosure can be implemented. The computing device 2000 may be implemented as or included in the source device 110 (or the video
15 encoder 114 or 200) or the destination device 120 (or the video decoder 124 or 300).

[00150] It would be appreciated that the computing device 2000 shown in Fig. 20 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.

[00151] As shown in Fig. 20, the computing device 2000 includes a general-purpose
20 computing device 2000. The computing device 2000 may at least comprise one or more processors or processing units 2010, a memory 2020, a storage unit 2030, one or more communication units 2040, one or more input devices 2050, and one or more output devices 2060.

[00152] In some embodiments, the computing device 2000 may be implemented as any
25 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
30 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 2000 can

5 support any type of interface to a user (such as “wearable” circuitry and the like).

[00153] The processing unit 2010 may be a physical or virtual processor and can implement various processes based on programs stored in the memory 2020. 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 2000.

10 The processing unit 2010 may also be referred to as a central processing unit (CPU), a microprocessor, a controller or a microcontroller.

[00154] The computing device 2000 typically includes various computer storage medium. Such medium can be any medium accessible by the computing device 2000, including, but not limited to, volatile and non-volatile medium, or detachable and non-detachable medium. The memory 2020 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 2030 may be any detachable or non-detachable medium and may include a machine-readable medium

15 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 2000.

[00155] The computing device 2000 may further include additional detachable/non-detachable, volatile/non-volatile memory medium. Although not shown in Fig. 20, 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.

25

[00156] The communication unit 2040 communicates with a further computing device via the communication medium. In addition, the functions of the components in the computing device 2000 can be implemented by a single computing cluster or multiple computing machines that can communicate via communication connections. Therefore,

30

the computing device 2000 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.

[00157] The input device 2050 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 2060 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 2040, the computing device 2000 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 2000, or any devices (such as a network card, a modem and the like) enabling the computing device 2000 to communicate with one or more other computing devices, if required. Such communication can be performed via input/output (I/O) interfaces (not shown).

[00158] In some embodiments, instead of being integrated in a single device, some or all components of the computing device 2000 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.

[00159] The computing device 2000 may be used to implement video

encoding/decoding in embodiments of the present disclosure. The memory 2020 may include one or more video coding modules 2025 having one or more program instructions. These modules are accessible and executable by the processing unit 2010 to perform the functionalities of the various embodiments described herein.

- 5 **[00160]** In the example embodiments of performing video encoding, the input device 2050 may receive video data as an input 2070 to be encoded. The video data may be processed, for example, by the video coding module 2025, to generate an encoded bitstream. The encoded bitstream may be provided via the output device 2060 as an output 2080.
- 10 **[00161]** In the example embodiments of performing video decoding, the input device 2050 may receive an encoded bitstream as the input 2070. The encoded bitstream may be processed, for example, by the video coding module 2025, to generate decoded video data. The decoded video data may be provided via the output device 2060 as the output 2080.
- 15 **[00162]** 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
- 20 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:

5 determining, for a conversion between a current video block of a video and a bitstream of the video, an affine candidate list of the current video block, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; and

performing the conversion based on the affine candidate list.

10 2. The method of claim 1, wherein the refined version of the affine candidate comprises a template matching refined affine candidate or a bilateral matching refined affine candidate.

15 3. The method of claim 1 or 2, wherein whether the refined and non-refined versions of the affine candidate are in the affine candidate list is based on a prediction direction of the affine candidate, the prediction direction comprising a bi-predicted direction or a uni-predicted direction.

20 4. The method of any of claims 1-3, wherein a plurality of affine candidates in a preliminary affine candidate list is checked based on an order, and if a current affine candidate in a first version satisfies a condition, a second version of the current affine candidate is used to append the affine candidate list.

25 5. The method of claim 4, wherein the order is based on a template matching cost.

6. The method of claim 4, wherein the condition indicates that the current affine candidate is coded with a uni-predicted direction.

30 7. The method of claim 4, wherein the current affine candidate is refined, and the second version of the current affine candidate comprises a non-refined version of the current affine candidate, or

wherein the current affine candidate is non-refined, and the second version of the current affine candidate comprises a refined version of the current affine candidate.

8. The method of any of claims 1-3, wherein a plurality of affine candidates in a preliminary affine candidate list is checked based on an order, and if a current affine candidate in a first version satisfies a condition, a second version of the current affine candidate is used to replace a target affine candidate in the affine candidate list.

5

9. The method of claim 8, wherein the target affine candidate is with a first index or after the first index.

10. The method of claim 9, wherein the first index is a constant value.

10

11. The method of claim 9, wherein an index of a first candidate with all-zero control point motion vectors is determined as the first index.

12. The method of claim 9, wherein the first index is based on the number of candidates of a type of the affine candidate.

15

13. The method of any of claims 1-12, wherein an ultimate index of an affine candidate in the affine candidate list is based on an initial index of the affine candidate and a variable.

20

14. The method of claim 13, wherein the initial index is parsed in the bitstream.

15. The method of claim 13 or 14, wherein the variable comprises a value in a range from 0 to a maximum number of allowed affine candidates in the affine candidate list.

25

16. The method of claim 13, wherein an index of a first candidate with all-zero control point motion vectors in the affine candidate list is determined as the variable.

17. The method of any of claims 13-16, wherein if the initial index of the affine candidate is smaller than the variable, the ultimate index of the affine candidate is the initial index.

30

18. The method of any of claims 13-17, wherein if the initial index of the affine candidate is larger than or equal to the variable, the ultimate index of the affine candidate is determined by adjusting the initial index at least in part based on the variable.

5 19. The method of claim 18, wherein the ultimate index is determined by subtracting the variable from the initial index.

10 20. The method of claim 18, wherein the ultimate index is determined based on at least one of: the initial index, the variable, or a prediction direction of the affine candidate.

21. The method of any of claims 13-20, wherein a version of the affine candidate specified by the ultimate index is used as an ultimate affine candidate.

15 22. The method of claim 21, wherein if the version of the affine candidate specified by the ultimate index is refined, a non-refined version of the affine candidate is used to generate an affine prediction.

20 23. The method of claim 21, wherein if the version of the affine candidate specified by the ultimate index is non-refined, a refined version of the affine candidate is used to generate an affine prediction.

24. The method of any of claims 1-23, wherein whether to and/or how to apply the method is based on a syntax element in the bitstream.

25 25. The method of claim 24, wherein the syntax element is at at least one of: a sequence level, a group of pictures level, a picture level, a slice level, or a tile group level.

30 26. The method of claim 24 or claim 25, wherein the syntax element is included in at least one of: a sequence header, a picture header, a sequence parameter set (SPS), a video parameter set (VPS), a decoded parameter set (DPS), decoding capability information (DCI), a picture parameter set (PPS), an adaptation parameter set (APS), a slice header, or a time group header.

27. The method of any of claims 24-26, wherein the syntax element is indicated in a region containing more than one sample or pixel.

28. The method of claim 27, wherein the region comprises one of: a prediction block (PB), a transform block (TB), a coding block (CB), a prediction unit (PU), a transform unit (TU), a coding unit (CU), a virtual pipeline data unit (VPDU), a coding tree unit (CTU), a CTU row, a slice, a tile, or a subpicture.

29. The method of any of claims 1-28, wherein whether to and/or how to apply the method is determined based on coding information of the current video block.

30. The method of claim 29, wherein the coding information comprises at least one of: a block size of the current video block, a color format of the current video block, a single or dual tree partitioning of the current video block, a color component of the current video block, a slice type of the current video block, or a picture type of the current video block.

31. The method of any of claims 1-30, wherein whether a first syntax element indicating if a template matching based refinement process is applied to a control point motion vector of the current video block is determined based on a second syntax element.

32. The method of any of claims 1-31, wherein the conversion includes encoding the current video block into the bitstream.

33. The method of any of claims 1-31, wherein the conversion includes decoding the current video block from the bitstream.

34. 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 1-33.

35. A non-transitory computer-readable storage medium storing instructions that cause a processor to perform a method in accordance with any of claims 1-33.

36. 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:

5 determining an affine candidate list of a current video block of the video, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate; and

generating the bitstream based on the affine candidate list.

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

10 determining an affine candidate list of a current video block of the video, the affine candidate list comprising a refined version of an affine candidate and a non-refined version of the affine candidate;

generating the bitstream based on the affine candidate list; and

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

15

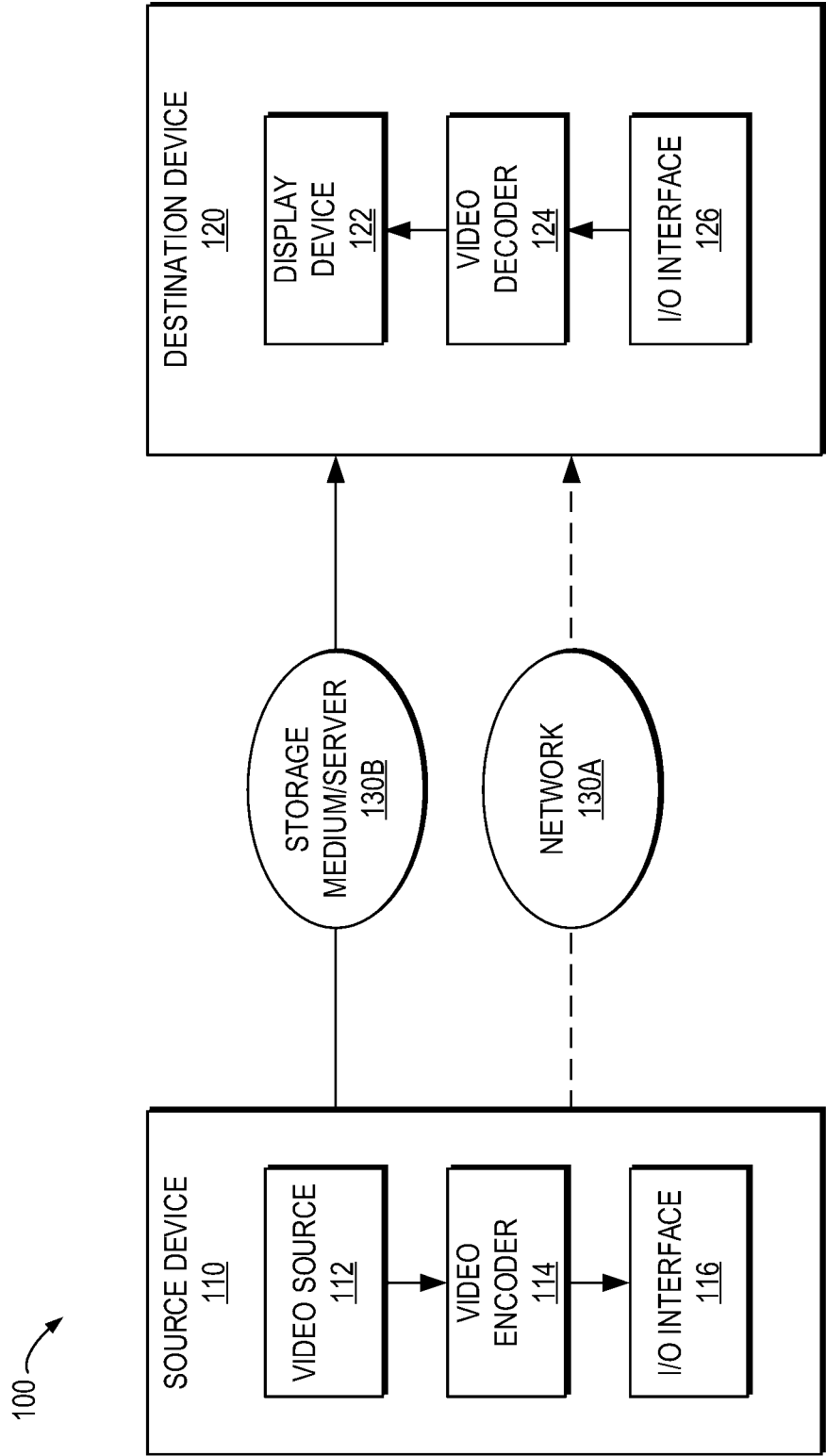


Fig. 1

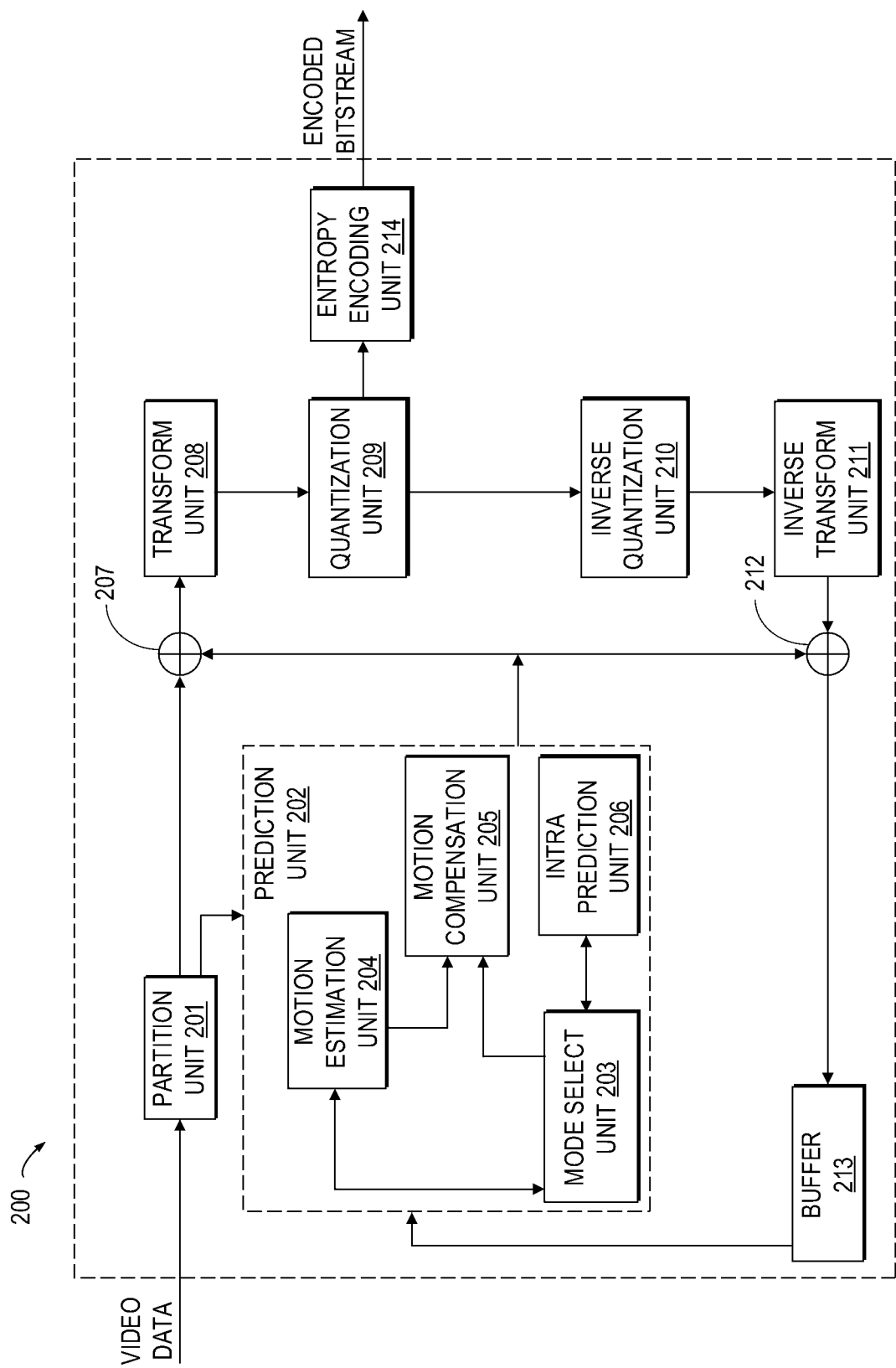


Fig. 2

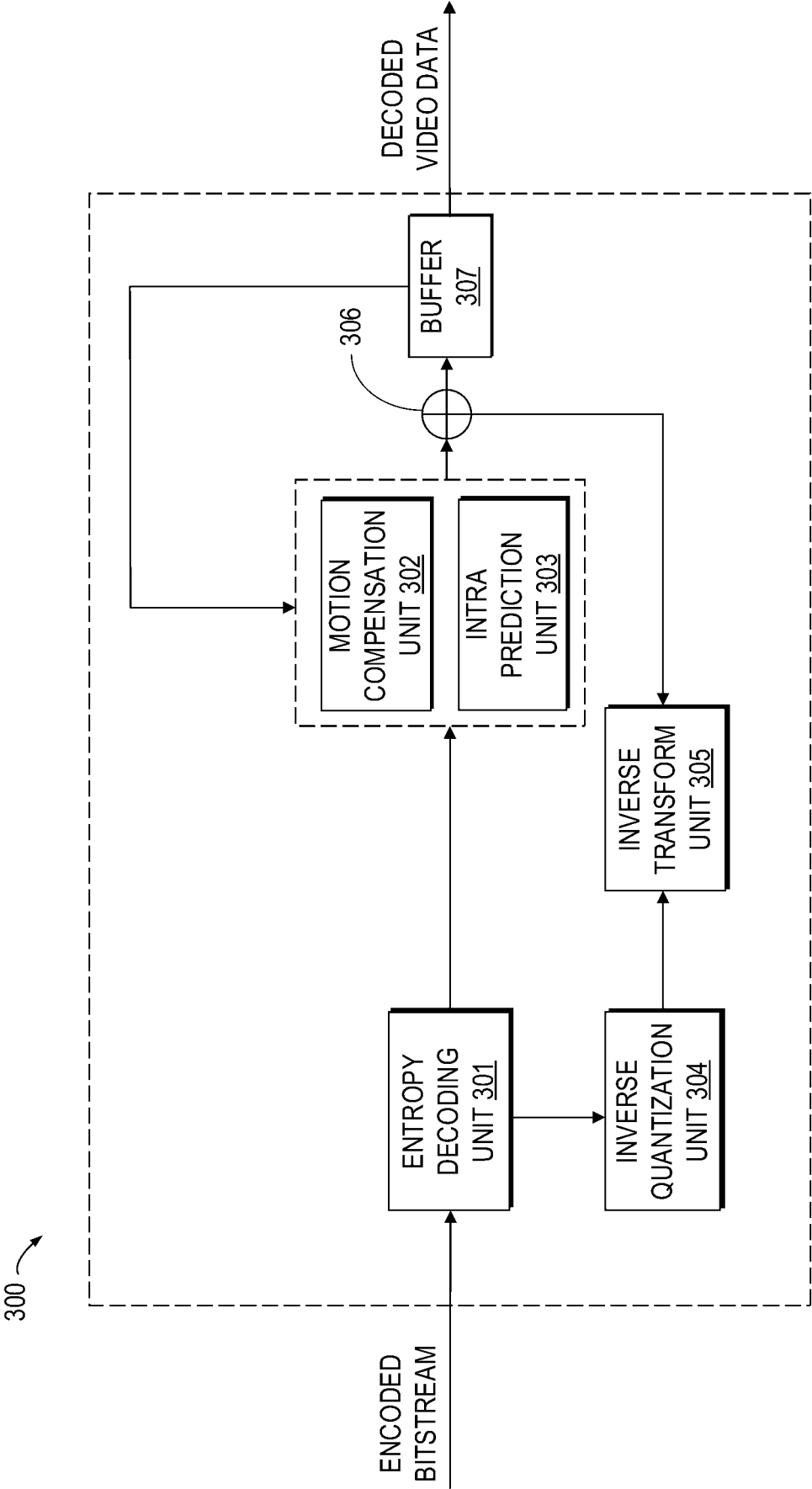


Fig. 3

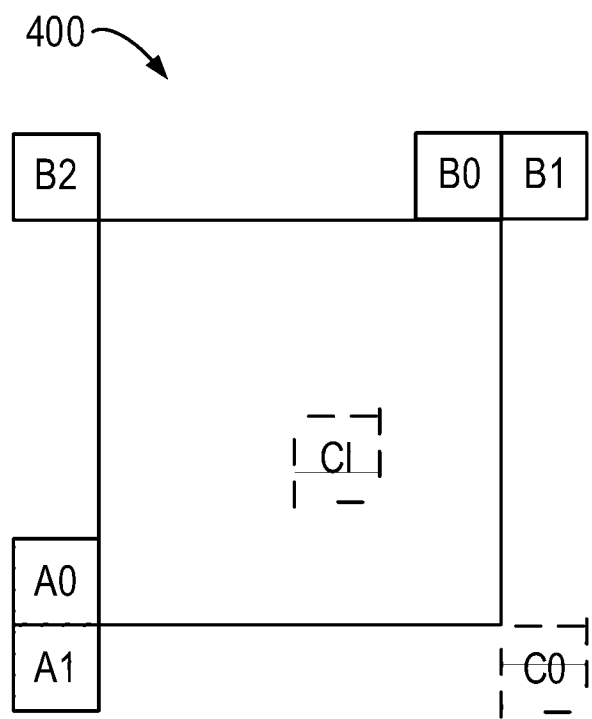


Fig. 4

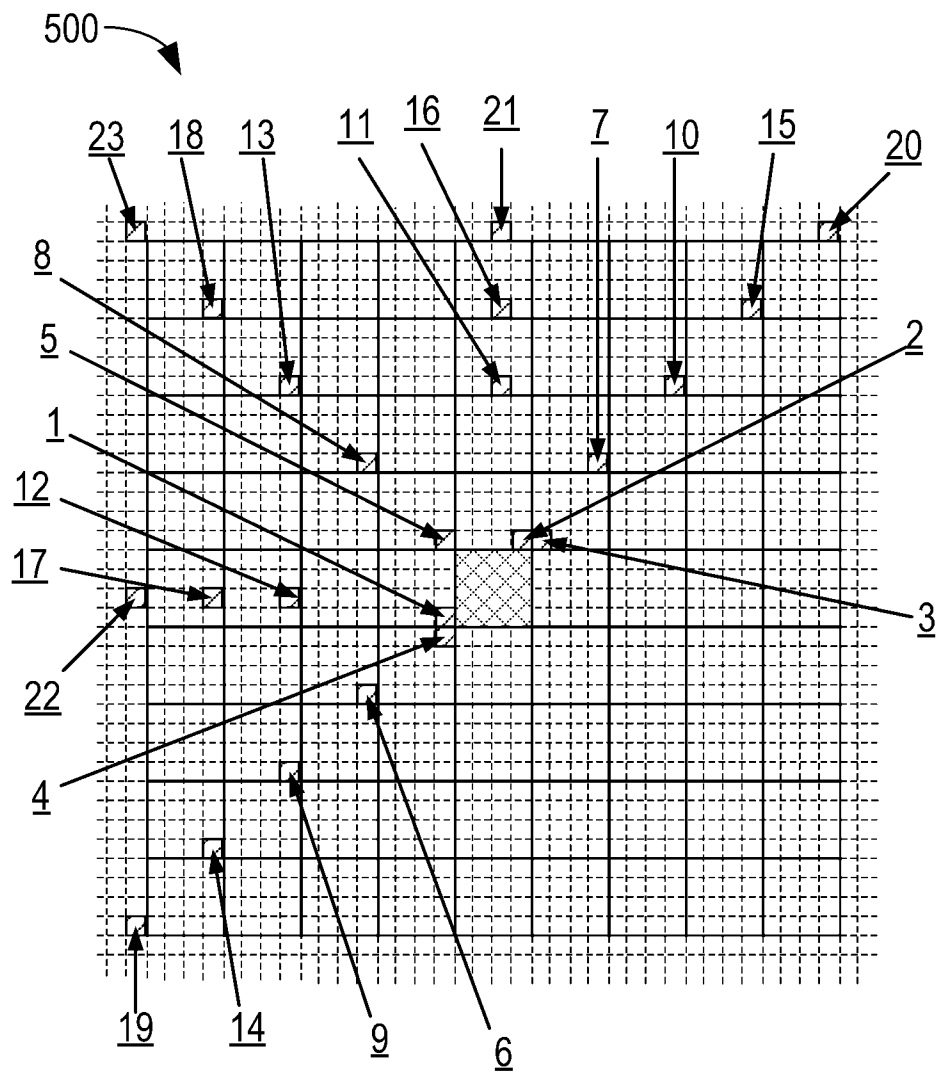


Fig. 5

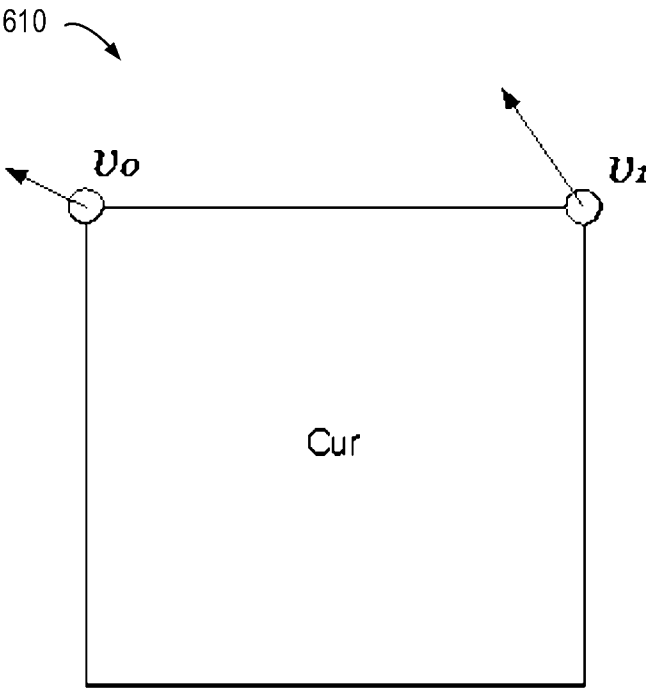


Fig. 6A

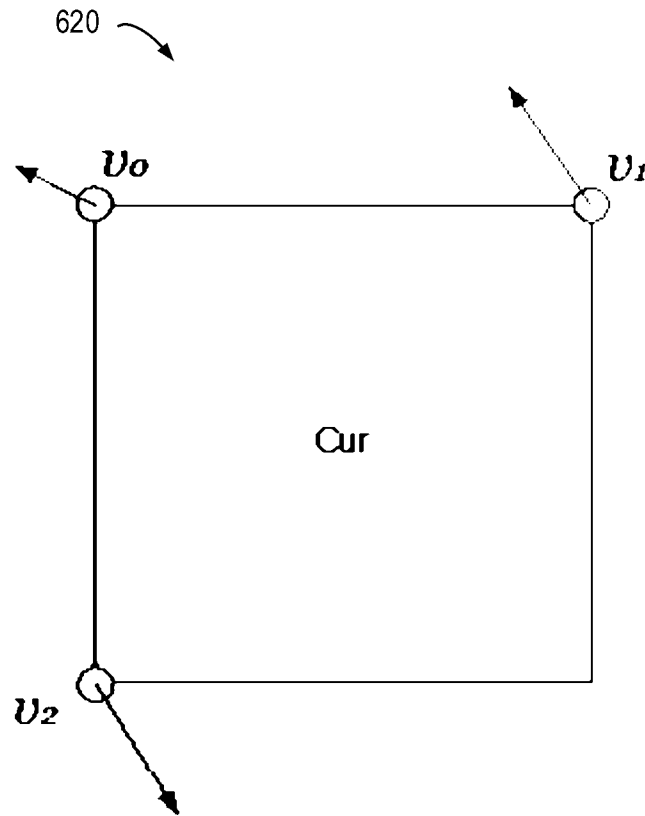


Fig. 6B

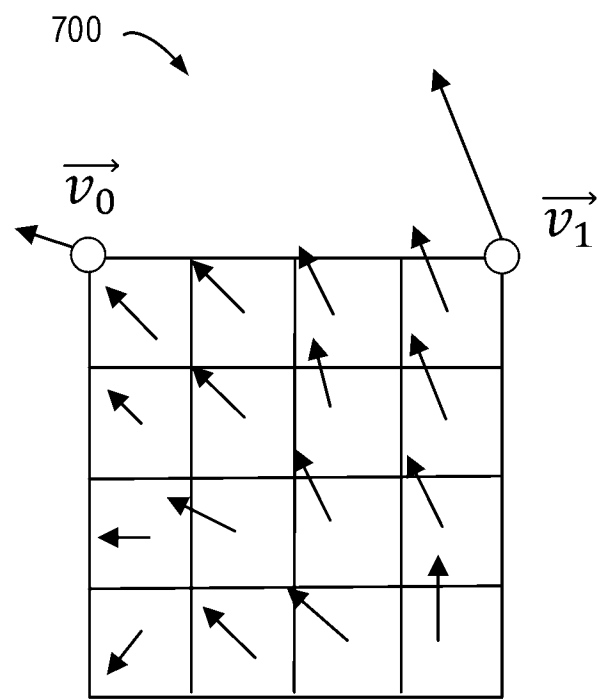


Fig. 7

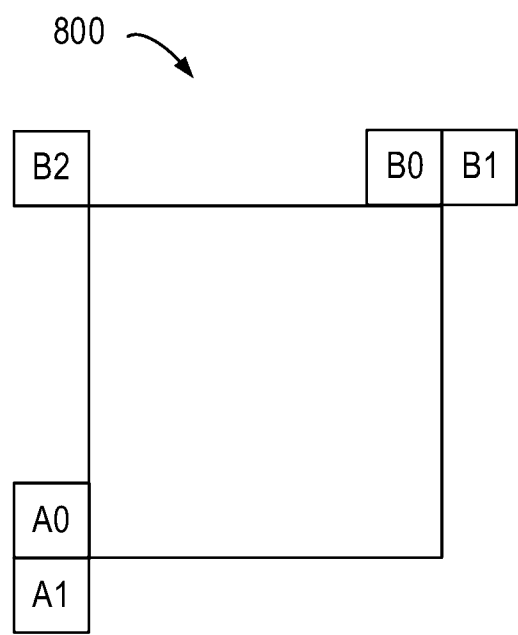


Fig. 8

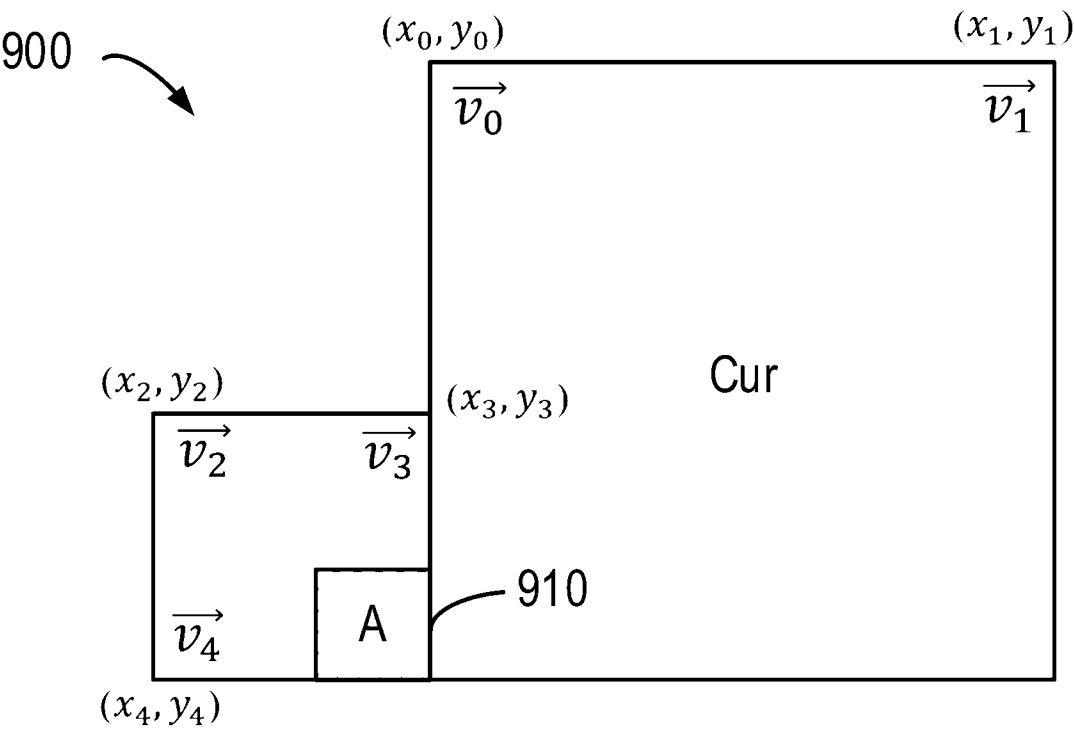


Fig. 9

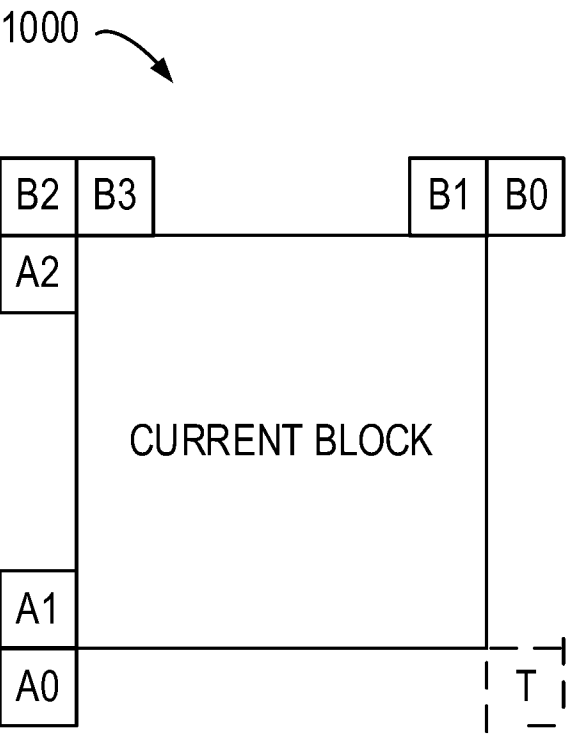


Fig. 10

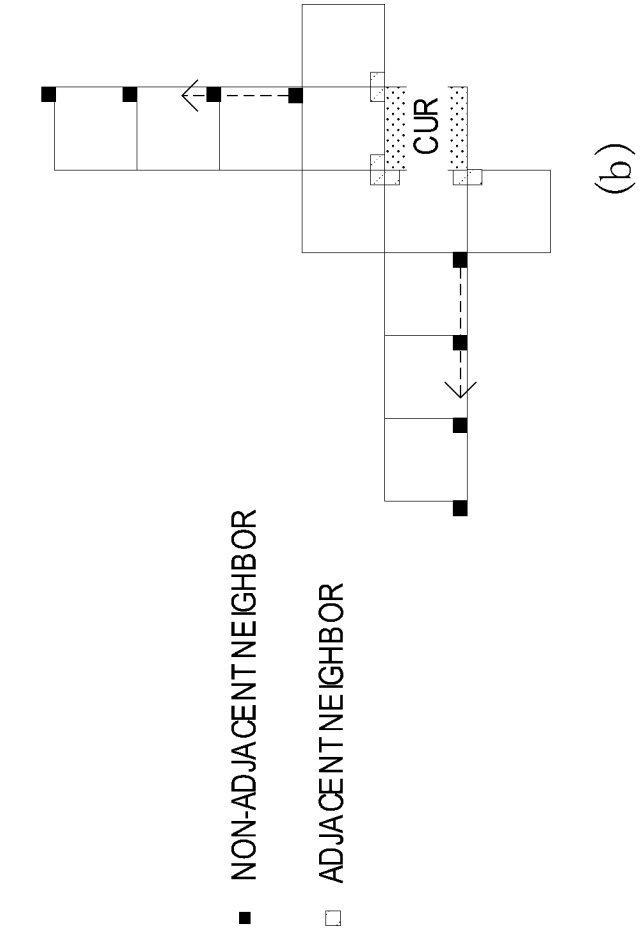


Fig. 11

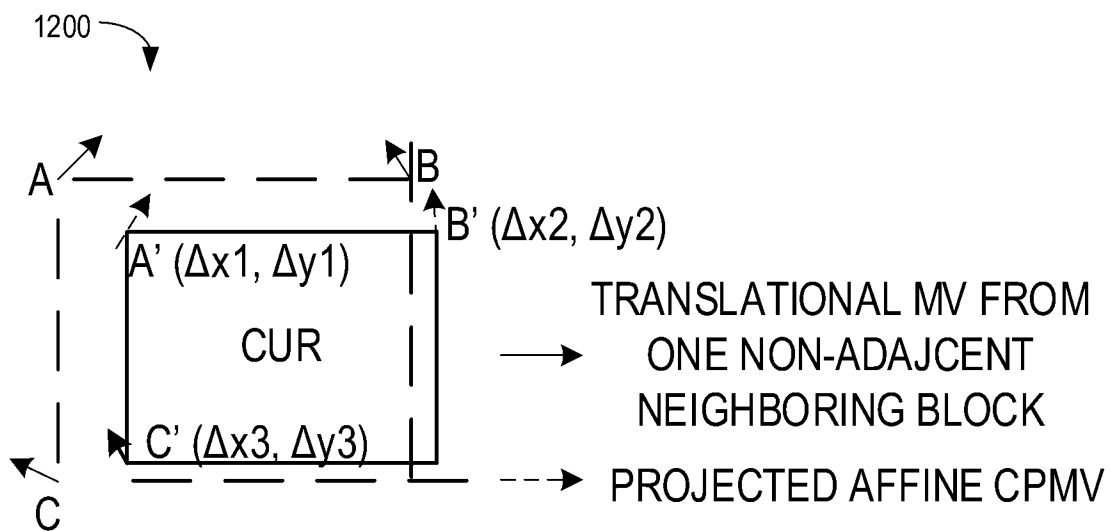


Fig. 12

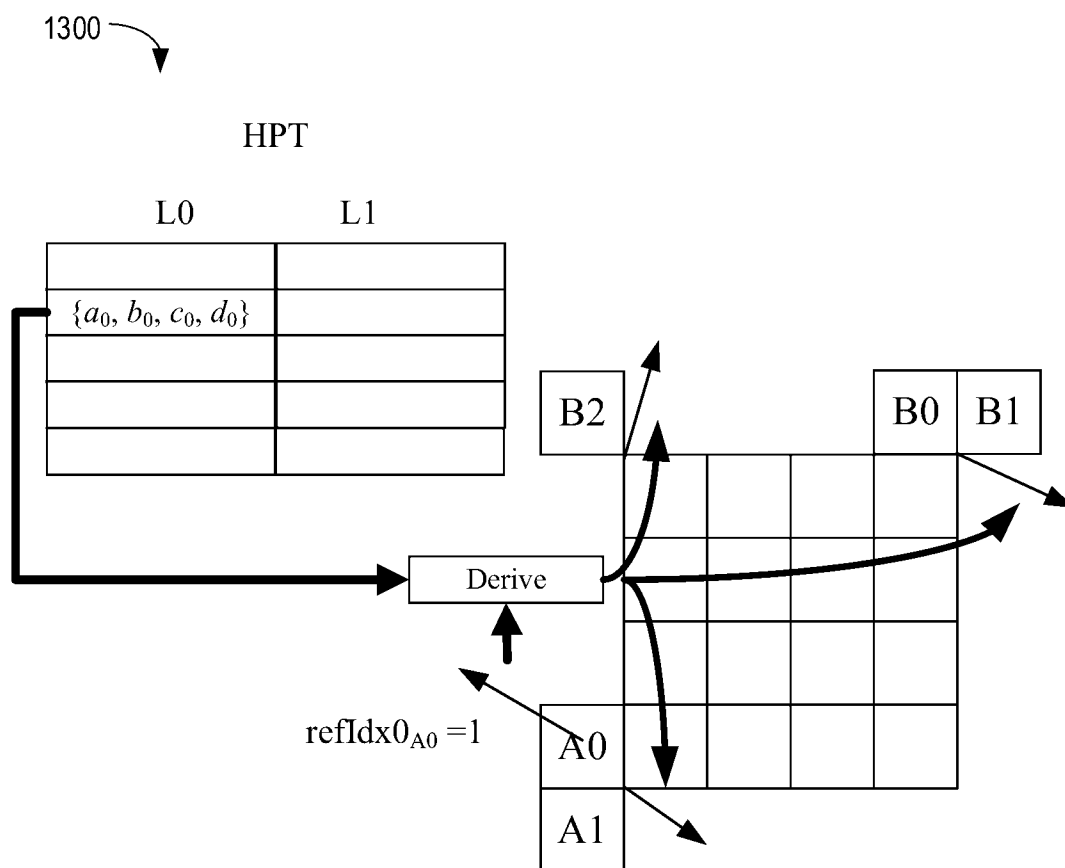


Fig. 13

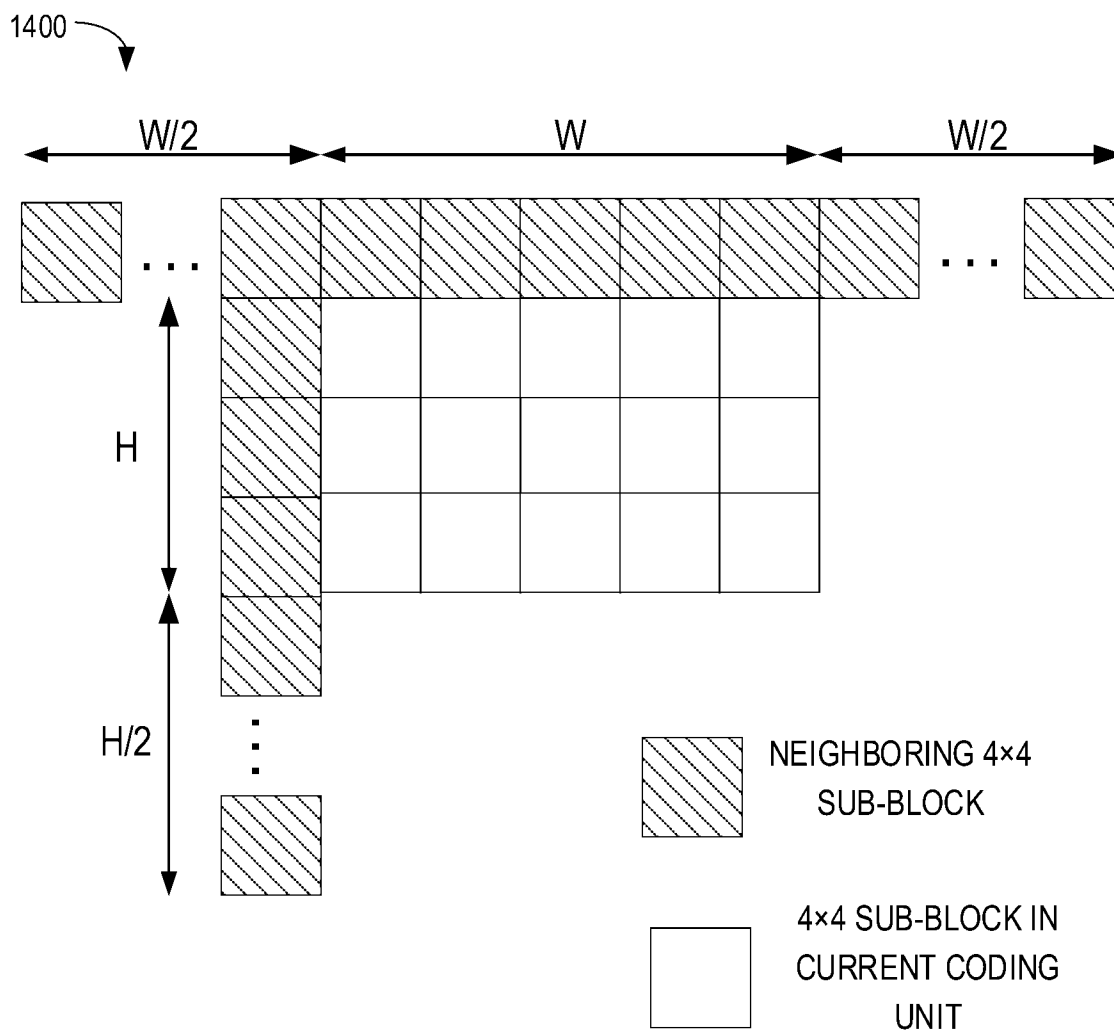


Fig. 14

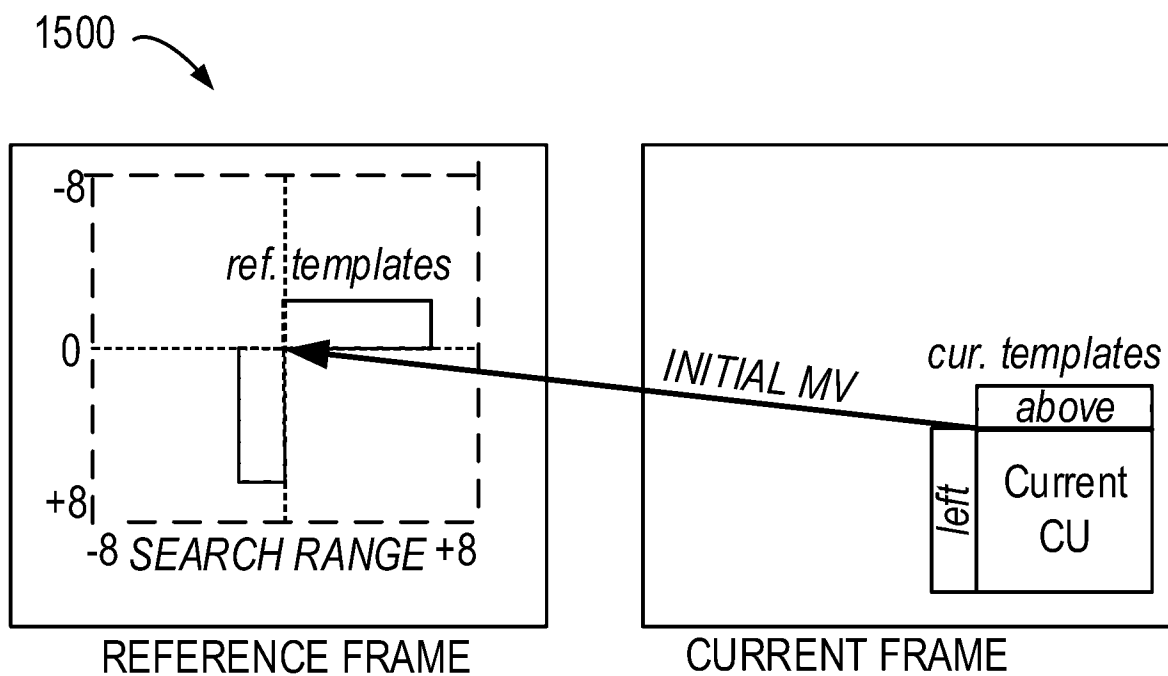


Fig. 15

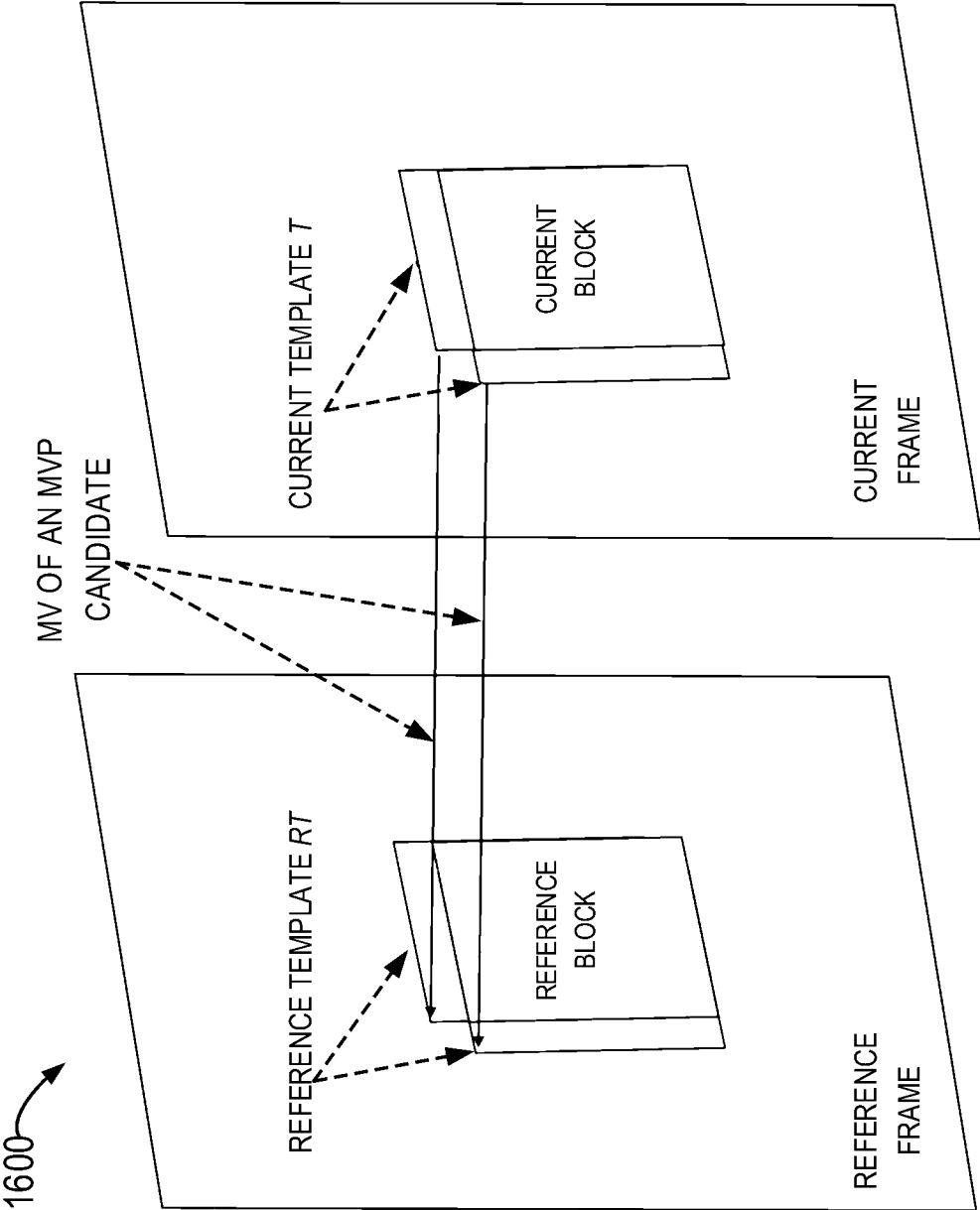


Fig. 16

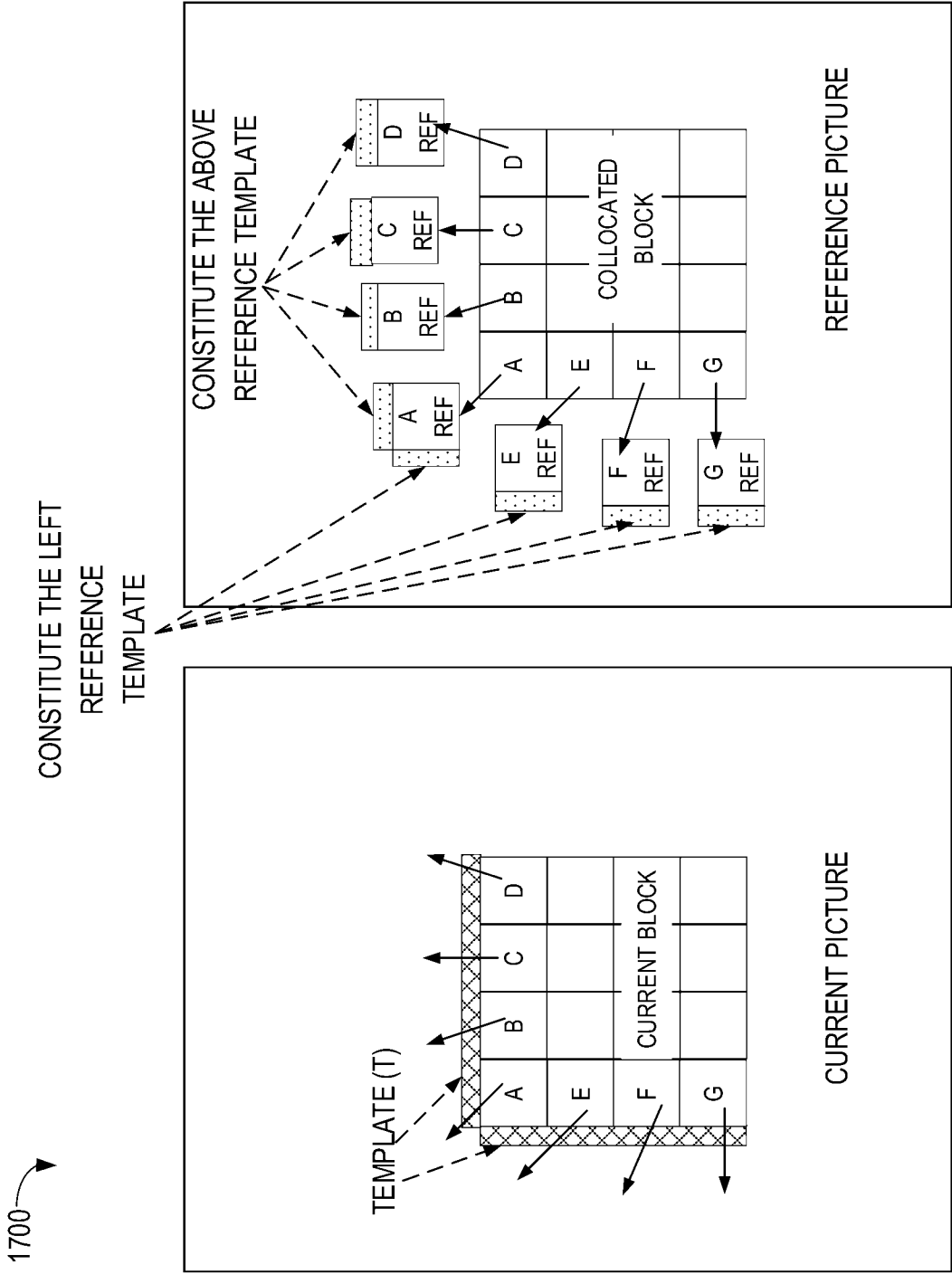


Fig. 17

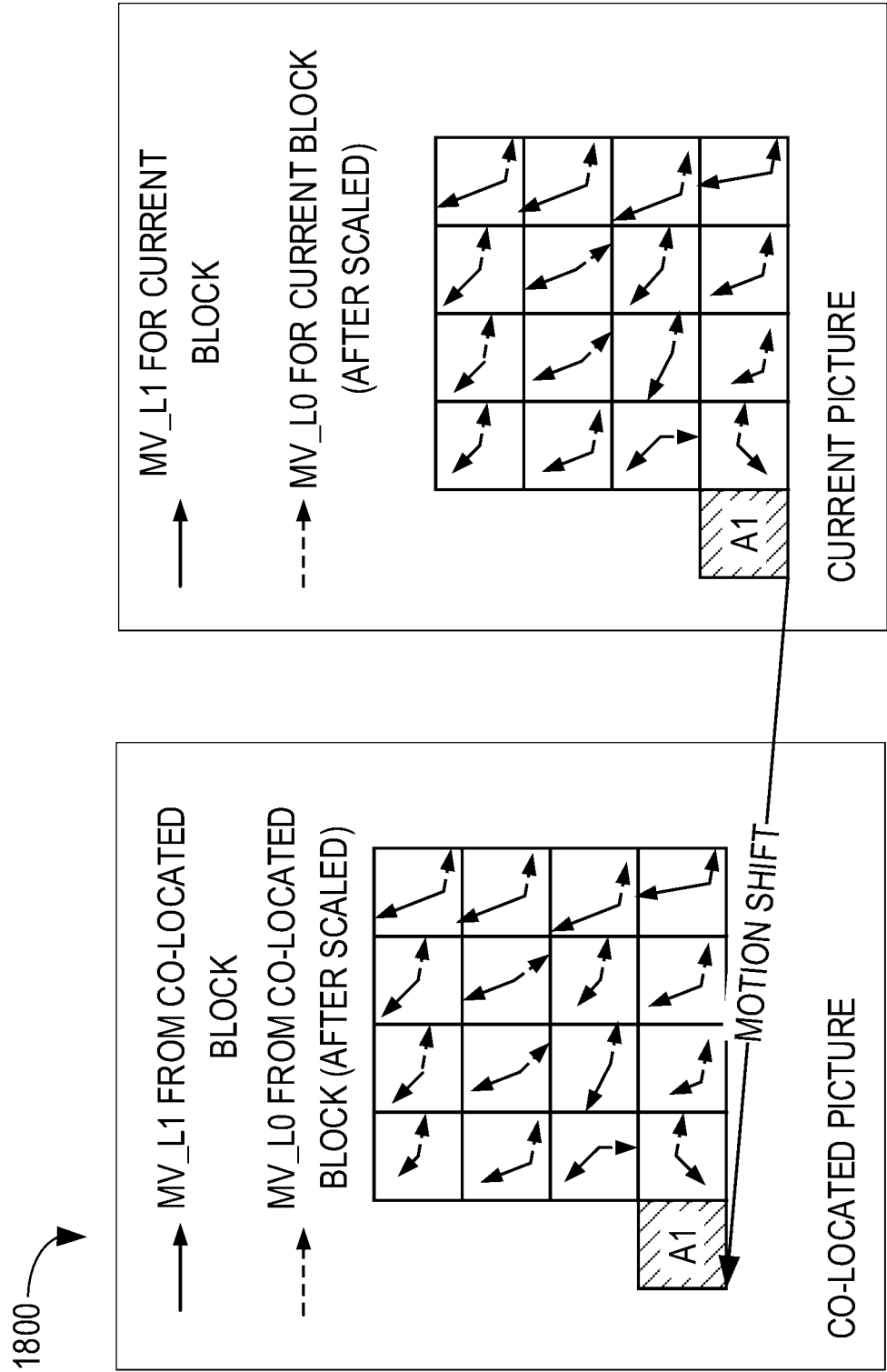


Fig. 18

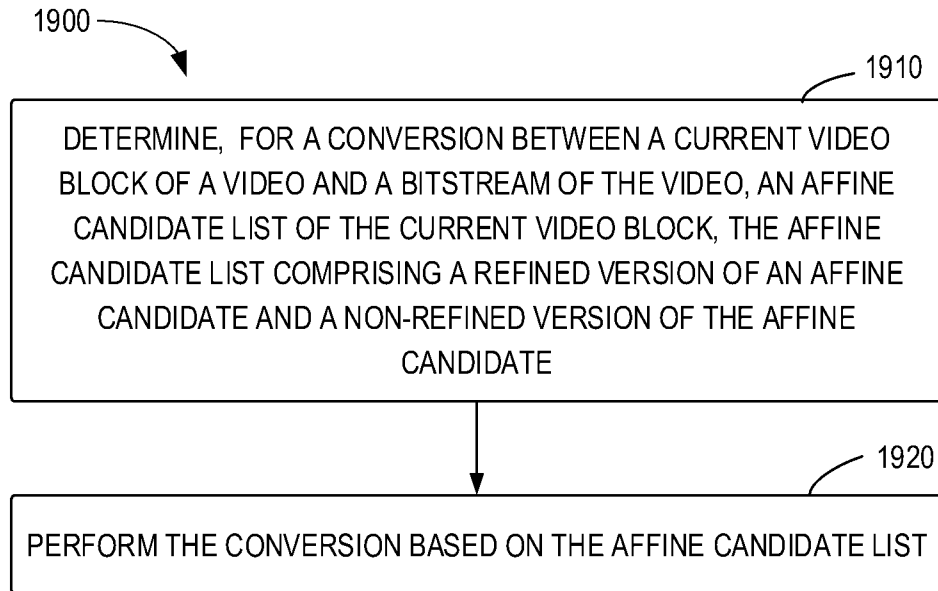


Fig. 19

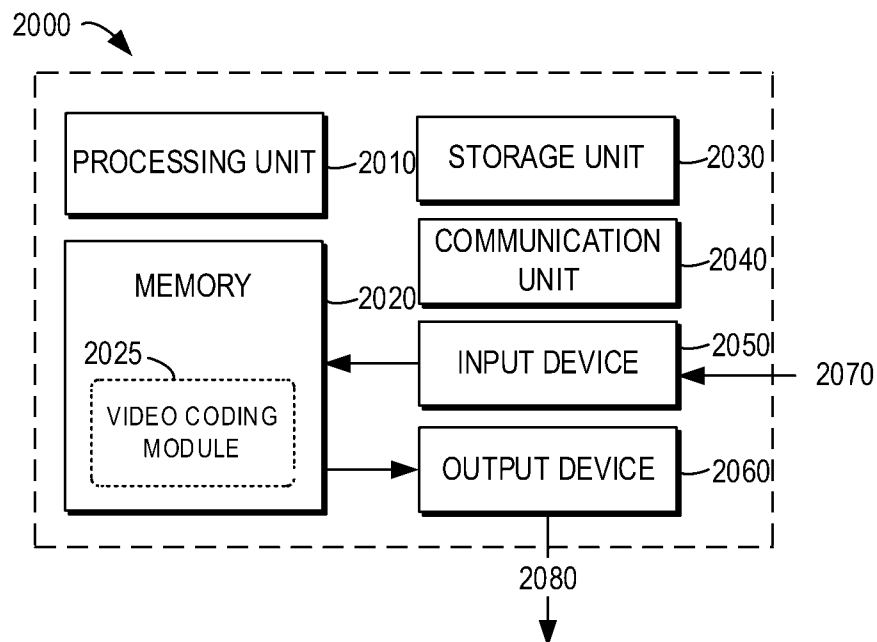


Fig. 20

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2024/084994

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/52(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)

CNABS,CNTXT,VEN,ENTXT,ENTXTC,CNKI:code, bitstream, affine, candidate, list, refine, index, template, bilateral, CPMV

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	CN 112956202 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO et al.) 11 June 2021 (2021-06-11) description, paragraphs [0030] to[0962], figures 1 to 64	1-37
A	CN 113170166 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO et al.) 23 July 2021 (2021-07-23) the whole document	1-37
A	CN 112385213 A (LG ELECTRONICS INC) 19 February 2021 (2021-02-19) the whole document	1-37
A	CN 113875234 A (TENCENT AMERICA LLC) 31 December 2021 (2021-12-31) the whole document	1-37
A	US 2021409722 A1 (BEIJING DAJIA INTERNET INFORMATION TECH CO LTD) 30 December 2021 (2021-12-30) the whole document	1-37

☐ 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

29 June 2024

Date of mailing of the international search report

01 July 2024

Name and mailing address of the ISA/CN

CHINA NATIONAL INTELLECTUAL PROPERTY
ADMINISTRATION
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing
100088, China

Authorized officer

DONG,ZhaoRu

Telephone No. (+86) 010-62411590

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2024/084994

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	112956202	A	11 June 2021	WO	2020094052	A1	14 May 2020
				WO	2020094054	A1	14 May 2020
				JP	2023011591	A	24 January 2023
				JP	7436597	B2	21 February 2024
				EP	4307678	A2	17 January 2024
				EP	4307678	A3	22 May 2024
				US	2021029372	A1	28 January 2021
				US	11166031	B2	02 November 2021
				US	2021029366	A1	28 January 2021
				US	11159808	B2	26 October 2021
				WO	2020094049	A1	14 May 2020
				US	2021006790	A1	07 January 2021
				US	11070820	B2	20 July 2021
				WO	2020094050	A1	14 May 2020
				US	2021337215	A1	28 October 2021
				US	11457226	B2	27 September 2022
				US	2021243438	A1	05 August 2021
				US	11611763	B2	21 March 2023
				US	2023283786	A1	07 September 2023
				US	2021266537	A1	26 August 2021
				US	11570450	B2	31 January 2023
				KR	20210087450	A	12 July 2021
				JP	2022506152	A	17 January 2022
				JP	7157246	B2	19 October 2022
				WO	2020094051	A1	14 May 2020
				US	2021006803	A1	07 January 2021
				US	11070821	B2	20 July 2021
				EP	3857890	A1	04 August 2021
				EP	3857890	A4	22 September 2021
CN	113170166	A	23 July 2021	US	2022132120	A1	28 April 2022
				US	11956431	B2	09 April 2024
				WO	2020140862	A1	09 July 2020
CN	112385213	A	19 February 2021	KR	20200140896	A	16 December 2020
				KR	102500091	B1	15 February 2023
				KR	20230022285	A	14 February 2023
				KR	102658929	B1	19 April 2024
				US	2023188728	A1	15 June 2023
				US	2021058628	A1	25 February 2021
				US	11606566	B2	14 March 2023
				WO	2019216714	A1	14 November 2019
				KR	20240055149	A	26 April 2024
CN	113875234	A	31 December 2021	CA	3131158	A1	06 October 2021
				JP	2024029127	A	05 March 2024
				WO	2021206992	A1	14 October 2021
				JP	2022532032	A	13 July 2022
				JP	7413401	B2	15 January 2024
				EP	3939265	A1	19 January 2022
				EP	3939265	A4	04 May 2022
				KR	20210145794	A	02 December 2021
				US	2022337856	A1	20 October 2022

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2024/084994

Patent document cited in search report				Publication date (day/month/year)		Patent family member(s)		Publication date (day/month/year)	
				US	12003748	B2	04 June 2024		
				AU	2023204371	A1	03 August 2023		
				AU	2021251670	A1	02 December 2021		
				AU	2021251670	B2	06 April 2023		
				US	2021314586	A1	07 October 2021		
				US	11405628	B2	02 August 2022		
				SG	11202110311	UA	29 November 2021		
US	2021409722	A1	30 December 2021	US	2021377558	A1	02 December 2021		
				US	11871024	B2	09 January 2024		
				EP	3928513	A1	29 December 2021		
				EP	3928513	A4	21 September 2022		
				US	11800140	B2	24 October 2023		
				US	2021377557	A1	02 December 2021		
				EP	3925222	A1	22 December 2021		
				EP	3925222	A4	10 August 2022		
				WO	2020172345	A1	27 August 2020		
				WO	2020172342	A1	27 August 2020		
				WO	2020172342	A9	25 March 2021		