



(51) International Patent Classification:

H04N 19/107 (2014.01)

H04N 19/182 (2014.01)

H04N 19/176 (2014.01)

(21) International Application Number:

PCT/CN2020/078988

(22) International Filing Date:

12 March 2020 (12.03.2020)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

PCT/CN2019/077838

12 March 2019 (12.03.2019)

CN

(71) Applicants: BEIJING BYTEDANCE NETWORK

TECHNOLOGY CO., LTD. [CN/CN]; Room B-0035, 2/F, No.3 Building, No.30, Shixing Road, Shijingshan District, Beijing 100041 (CN). BYTEDANCE INC. [US/US];

12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US).

(72) Inventors: ZHU, Weijia; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). XU, Jizheng; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). ZHANG, Li; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). ZHANG, Kai; 12655 West Jefferson Boulevard, Sixth Floor, Suite No. 137, Los Angeles, California 90066 (US). LIU, Hongbin; Jinritoutiao Post Office, China Satellite Communications Tower, No.63, Zhichun Road, Haidian District, Beijing 100080 (CN). WANG, Yue; Jinritoutiao Post Office, China Satellite Communications Tower, No.63, Zhichun Road, Haidian District, Beijing 100080 (CN).

(74) Agent: LIU, SHEN & ASSOCIATES; 10th Floor, Building 1, 10 Caihefang Road, Haidian District, Beijing 100080 (CN).

(54) Title: ADAPTIVE WEIGHT IN MULTI-HYPOTHESIS PREDICTION IN VIDEO CODING

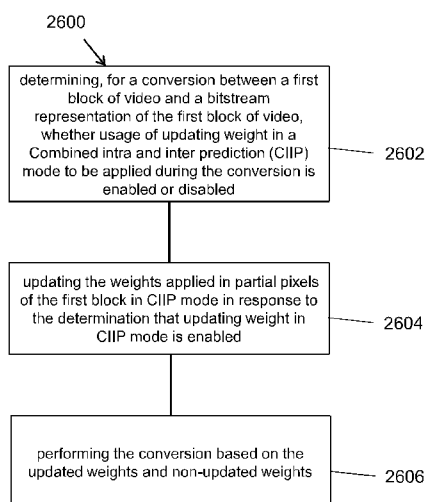


FIG. 26

(57) Abstract: Adaptive weight in multi-hypothesis prediction in video coding is disclosed. In one example implementations, determining, for a conversion between a first block of video and a bitstream representation of the first block of video, whether usage of updating weight in a Combined intra and inter prediction (CIIP) mode to be applied during the conversion is enabled or disabled; updating the weights applied in partial pixels of the first block in CIIP mode in response to the determination that updating weight in CIIP mode is enabled; and performing the conversion based on the updated weights and non-updated weights.

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

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

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

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

ADAPTIVE WEIGHT IN MULTI-HYPOTHESIS PREDICTION IN VIDEO CODING

CROSS-REFERENCE TO RELATED APPLICATION

[0001] Under the applicable patent law and/or rules pursuant to the Paris Convention, this application is made to timely claim the priority to and benefits of International Patent Application No. PCT/CN2019/077838, filed on March 12, 2019. The entire disclosures of International Patent Application No. PCT/CN2019/077838 is incorporated by reference as part of the disclosure of this application.

TECHNICAL FIELD

[0002] This patent document relates to video coding techniques, devices and systems.

BACKGROUND

[0003] In spite of the advances in video compression, digital video still accounts for the largest bandwidth use on the internet and other digital communication networks. As the number of connected user devices capable of receiving and displaying video increases, it is expected that the bandwidth demand for digital video usage will continue to grow.

SUMMARY

[0004] The present document describes various embodiments and techniques for using multi-hypothesis prediction in video coding.

[0005] In one example aspect, various methods for using multi-hypothesis prediction in video coding are disclosed.

[0006] In another example aspect, a method of video processing is disclosed. The method includes updating, during a conversion between a current video region and a bitstream representation of the current video block, weights used for a combined inter and intra prediction mode and performing the conversion based on the updating.

[0007] In another example aspect, a method of video processing is disclosed. The method includes determining, for a conversion between a first block of video and a bitstream representation of the first block of video, whether usage of updating weight in a Combined intra

and inter prediction (CIIP) mode to be applied during the conversion is enabled or disabled; updating the weights applied in partial pixels of the first block in CIIP mode in response to the determination that updating weight in CIIP mode is enabled; and performing the conversion based on the updated weights and non-updated weights.

[0008] In another example aspect, a method of video processing is disclosed. The method includes determining, for a conversion between a first block of video and a bitstream representation of the first block of video, a weight set from multiple weight sets being used in a Combined intra and inter prediction (CIIP) mode depends on a message presented in at least one of the following: sequence level including sequence parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture header, and block level including coding tree unit (CTU) and coding unit (CU); applying CIIP mode to generate a final prediction of the first block based on the determined weight set; and performing the conversion based on the final prediction; wherein the final prediction of the first block is generated based on a weighted sum of an intermediate intra prediction and an intermediate merge inter prediction of the first block.

[0009] In yet another example aspect, a video encoder apparatus configured to implement one of the above-described methods is disclosed.

[0010] In yet another example aspect, a video decoder apparatus configured to implement one of the above-described methods is disclosed.

[0011] In yet another aspect, a computer-readable medium is disclosed. A processor-executable code for implementing one of the above-described methods is stored on the computer-readable medium.

[0012] These, and other aspects, are described in the present document.

BRIEF DESCRIPTION OF DRAWINGS

[0013] FIG. 1 shows an example derivation process for merge candidates list construction.

[0014] FIG. 2 shows examples of positions of spatial merge candidates.

[0015] FIG. 3 shows examples of candidate pairs considered for redundancy check of spatial merge candidates.

[0016] FIG. 4 shows examples of positions for the second PU of $N \times 2N$ and $2N \times N$ partitions.

[0017] FIG. 5 is an example illustration of motion vector scaling for temporal merge

candidate.

[0018] FIG. 6 shows examples of candidate positions for temporal merge candidate, C0 and C1.

[0019] FIG. 7 shows an example of combined bi-predictive merge candidate.

[0020] FIG. 8 summarizes derivation process for motion vector prediction candidate.

[0021] FIG. 9 shows an illustration of motion vector scaling for spatial motion vector candidate.

[0022] FIG. 10 shows neighbouring samples used for deriving IC parameters

[0023] FIG. 11 shows a simplified affine motion model for (a) 4-parameter affine (b) 6-parameter affine.

[0024] FIG. 12 Example of Affine MVF per sub-block.

[0025] FIG. 13 shows a 4-parameter affine model (a) and 6-parameter affine model (b).

[0026] FIG. 14 shows MVP for AF_INTER for inherited affine candidates.

[0027] FIG. 15 shows MVP for AF_INTER for constructed affine candidates.

[0028] FIG. 16 shows candidates for AF_merge.

[0029] FIG. 17 shows Candidates position for affine merge mode.

[0030] FIG. 18 shows an example of MMVD search process.

[0031] FIG. 19 shows an example of MMVD search point.

[0032] FIG. 20 shows DMVR based on bilateral template matching.

[0033] FIG. 21 is an example of MVD (0, 1) mirrored between list 0 and list 1 in DMVR.

[0034] FIG. 22 shows MVs that may be checked in one iteration.

[0035] FIG. 23 shows an example of required reference samples with padding.

[0036] FIG. 24 shows an example apparatus for implementing a technique described in the present document.

[0037] FIG. 25 is a flowchart for an example method of video processing.

[0038] FIG. 26 is a flowchart for an example method of video processing.

[0039] FIG. 27 is a flowchart for an example method of video processing.

[0040] In the description of drawings (a) and (b) refer to the left and right hand side of the corresponding drawing.

DETAILED DESCRIPTION

[0041] The present document provides various techniques that can be used by a decoder of image or video bitstreams to improve the quality of decompressed or decoded digital video or images. For brevity, the term “video” is used herein to include both a sequence of pictures (traditionally called video) and individual images. Furthermore, a video encoder may also implement these techniques during the process of encoding in order to reconstruct decoded frames used for further encoding.

[0042] Section headings are used in the present document for ease of understanding and do not limit the embodiments and techniques to the corresponding sections. As such, embodiments from one section can be combined with embodiments from other sections.

[0043] 1. Brief Summary

[0044] This patent document is related to video coding technologies. Specifically, it is related to candidate list construction in video coding. It may be applied to the existing video coding standard like HEVC, or the standard (Versatile Video Coding) to be finalized. It may be also applicable to future video coding standards or video codec.

[0045] 2. Initial Discussion

[0046] Video coding standards have evolved primarily through the development of the well-known ITU-T and ISO/IEC standards. The ITU-T produced H.261 and H.263, ISO/IEC produced MPEG-1 and MPEG-4 Visual, and the two organizations jointly produced the H.262/MPEG-2 Video and H.264/MPEG-4 Advanced Video Coding (AVC) and H.265/HEVC standards. Since H.262, the video coding standards are based on the hybrid video coding structure wherein temporal prediction plus transform coding are utilized. To explore the future video coding technologies beyond HEVC, Joint Video Exploration Team (JVET) was founded by VCEG and MPEG jointly in 2015. Since then, many new methods have been adopted by JVET and put into the reference software named Joint Exploration Model (JEM). In April 2018, the Joint Video Expert Team (JVET) between VCEG (Q6/16) and ISO/IEC JTC1 SC29/WG11 (MPEG) was created to work on the VVC standard targeting at 50% bitrate reduction compared to HEVC.

[0047] 2.1 Inter prediction in HEVC/H.265

[0048] Each inter-predicted PU has motion parameters for one or two reference picture lists. Motion parameters include a motion vector and a reference picture index. Usage of one of the

two reference picture lists may also be signalled using `inter_pred_idc`. Motion vectors may be explicitly coded as deltas relative to predictors.

[0049] When a CU is coded with skip mode, one PU is associated with the CU, and there are no significant residual coefficients, no coded motion vector delta or reference picture index. A merge mode is specified whereby the motion parameters for the current PU are obtained from neighbouring PUs, including spatial and temporal candidates. The merge mode can be applied to any inter-predicted PU, not only for skip mode. The alternative to merge mode is the explicit transmission of motion parameters, where motion vector (to be more precise, motion vector differences (MVD) compared to a motion vector predictor), corresponding reference picture index for each reference picture list and reference picture list usage are signalled explicitly per each PU. Such a mode is named Advanced motion vector prediction (AMVP) in this disclosure.

[0050] When signalling indicates that one of the two reference picture lists is to be used, the PU is produced from one block of samples. This is referred to as 'uni-prediction'. Uni-prediction is available both for P-slices and B-slices.

[0051] When signalling indicates that both of the reference picture lists are to be used, the PU is produced from two blocks of samples. This is referred to as 'bi-prediction'. Bi-prediction is available for B-slices only.

[0052] The following text provides the details on the inter prediction modes specified in HEVC. The description will start with the merge mode.

[0053] 2.1.1 Reference picture list

[0054] In HEVC, the term inter prediction is used to denote prediction derived from data elements (e.g., sample values or motion vectors) of reference pictures other than the current decoded picture. Like in H.264/AVC, a picture can be predicted from multiple reference pictures. The reference pictures that are used for inter prediction are organized in one or more reference picture lists. The reference index identifies which of the reference pictures in the list should be used for creating the prediction signal.

[0055] A single reference picture list, List 0, is used for a P slice and two reference picture lists, List 0 and List 1 are used for B slices. It should be noted reference pictures included in List 0/1 could be from past and future pictures in terms of capturing/display order.

[0056] 2.1.2 Merge Mode in HEVC

[0057] 2.1.2.1 Derivation of candidates for merge mode

[0058] When a PU is predicted using merge mode, an index pointing to an entry in the merge candidates list is parsed from the bitstream and used to retrieve the motion information. The construction of this list is specified in the HEVC standard and can be summarized according to the following sequence of steps:

[0059] Step 1: Initial candidates derivation

[0060] Step 1.1: Spatial candidates derivation

[0061] Step 1.2: Redundancy check for spatial candidates

[0062] Step 1.3: Temporal candidates derivation

[0063] Step 2: Additional candidates insertion

[0064] Step 2.1: Creation of bi-predictive candidates

[0065] Step 2.2: Insertion of zero motion candidates

[0066] These steps are also schematically depicted in FIG. 1. For spatial merge candidate derivation, a maximum of four merge candidates are selected among candidates that are located in five different positions. For temporal merge candidate derivation, a maximum of one merge candidate is selected among two candidates. Since constant number of candidates for each PU is assumed at decoder, additional candidates are generated when the number of candidates obtained from step 1 does not reach the maximum number of merge candidate (MaxNumMergeCand) which is signalled in slice header. Since the number of candidates is constant, index of best merge candidate is encoded using truncated unary binarization (TU). If the size of CU is equal to 8, all the PUs of the current CU share a single merge candidate list, which is identical to the merge candidate list of the $2N \times 2N$ prediction unit.

[0067] In the following, the operations associated with the aforementioned steps are detailed.

[0068] **2.1.2.2 Spatial candidates derivation**

[0069] In the derivation of spatial merge candidates, a maximum of four merge candidates are selected among candidates located in the positions depicted in FIG. 2. The order of derivation is A1, B1, B0, A0 and B2. Position B2 is considered only when any PU of position A1, B1, B0, A0 is not available (e.g. because it belongs to another slice or tile) or is intra coded. After candidate at position A1 is added, the addition of the remaining candidates is subject to a redundancy check which ensures that candidates with same motion information are excluded from the list so that coding efficiency is improved. To reduce computational complexity, not all possible candidate pairs are considered in the mentioned redundancy check. Instead only the

pairs linked with an arrow in FIG. 3 are considered and a candidate is only added to the list if the corresponding candidate used for redundancy check has not the same motion information.

Another source of duplicate motion information is the “second PU” associated with partitions different from $2N \times 2N$. As an example, FIG. 4 depicts the second PU for the case of $N \times 2N$ and $2N \times N$, respectively. When the current PU is partitioned as $N \times 2N$, candidate at position A1 is not considered for list construction. In fact, by adding this candidate will lead to two prediction units having the same motion information, which is redundant to just have one PU in a coding unit.

Similarly, position B1 is not considered when the current PU is partitioned as $2N \times N$.

[0070] 2.1.2.3 Temporal candidates derivation

[0071] In this step, only one candidate is added to the list. Particularly, in the derivation of this temporal merge candidate, a scaled motion vector is derived based on co-located PU belonging to the picture which has the smallest POC difference with current picture within the given reference picture list. The reference picture list to be used for derivation of the co-located PU is explicitly signalled in the slice header. The scaled motion vector for temporal merge candidate is obtained as illustrated by the dotted line in FIG. 5, which is scaled from the motion vector of the co-located PU using the POC distances, t_b and t_d , where t_b is defined to be the POC difference between the reference picture of the current picture and the current picture and t_d is defined to be the POC difference between the reference picture of the co-located picture and the co-located picture. The reference picture index of temporal merge candidate is set equal to zero. A practical realization of the scaling process is described in the HEVC specification. For a B-slice, two motion vectors, one is for reference picture list 0 and the other is for reference picture list 1, are obtained and combined to make the bi-predictive merge candidate.

[0072] In the co-located PU (Y) belonging to the reference frame, the position for the temporal candidate is selected between candidates C0 and C1, as depicted in FIG. 6. If PU at position C0 is not available, is intra coded, or is outside of the current coding tree unit (CTU aka. LCU, largest coding unit) row, position C1 is used. Otherwise, position C0 is used in the derivation of the temporal merge candidate.

[0073] Temporal Motion Vector Prediction is also known as “TMVP”.

[0074] 2.1.2.4 Additional candidates insertion

[0075] Besides spatial and temporal merge candidates, there are two additional types of merge candidates: combined bi-predictive merge candidate and zero merge candidate. Combined

bi-predictive merge candidates are generated by utilizing spatial and temporal merge candidates. Combined bi-predictive merge candidate is used for B-Slice only. The combined bi-predictive candidates are generated by combining the first reference picture list motion parameters of an initial candidate with the second reference picture list motion parameters of another. If these two tuples provide different motion hypotheses, they will form a new bi-predictive candidate. As an example, FIG. 7 depicts the case when two candidates in the original list (on the left), which have $mvL0$ and $refIdxL0$ or $mvL1$ and $refIdxL1$, are used to create a combined bi-predictive merge candidate added to the final list (on the right). There are numerous rules regarding the combinations which are considered to generate these additional merge candidates.

[0076] Zero motion candidates are inserted to fill the remaining entries in the merge candidates list and therefore hit the $MaxNumMergeCand$ capacity. These candidates have zero spatial displacement and a reference picture index which starts from zero and increases every time a new zero motion candidate is added to the list. Finally, no redundancy check is performed on these candidates.

[0077] 2.1.3 AMVP

[0078] AMVP exploits spatio-temporal correlation of motion vector with neighbouring PUs, 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 neighbouring PU positions, removing redundant candidates and adding zero vector to make the candidate list to be constant length. Then, the encoder can select the best predictor from the candidate list and transmit the corresponding index indicating the chosen candidate. Similarly with merge index signalling, the index of the best motion vector candidate is encoded using truncated unary. The maximum value to be encoded in this case is 2 (see FIG. 8). In the following sections, details about derivation process of motion vector prediction candidate are provided.

[0079] 2.1.3.1 Derivation of AMVP candidates

[0080] FIG. 8 shows an example derivation process for motion vector prediction candidates.

[0081] In motion vector prediction, two types of motion vector candidates are considered: spatial motion vector candidate and temporal motion vector candidate. For spatial motion vector candidate derivation, two motion vector candidates are eventually derived based on motion vectors of each PU located in five different positions as depicted in FIG. 2.

[0082] For temporal motion vector candidate derivation, one motion vector candidate is selected from two candidates, which are derived based on two different co-located positions. After the first list of spatio-temporal candidates is made, duplicated motion vector candidates in the list are removed. If the number of potential candidates is larger than two, motion vector candidates whose reference picture index within the associated reference picture list is larger than 1 are removed from the list. If the number of spatio-temporal motion vector candidates is smaller than two, additional zero motion vector candidates is added to the list.

[0083] 2.1.3.2 Spatial motion vector candidates

[0084] In the derivation of spatial motion vector candidates, a maximum of two candidates are considered among five potential candidates, which are derived from PUs located in positions as depicted in FIG. 2, those positions being the same as those of motion merge. The order of derivation for the left side of the current PU is defined as A0, A1, and scaled A0, scaled A1. The order of derivation for the above side of the current PU is defined as B0, B1, B2, scaled B0, scaled B1, scaled B2. For each side there are therefore four cases that can be used as motion vector candidate, with two cases not required to use spatial scaling, and two cases where spatial scaling is used. The four different cases are summarized as follows.

[0085] No spatial scaling

[0086] (1) Same reference picture list, and same reference picture index (same POC)

[0087] (2) Different reference picture list, but same reference picture (same POC)

[0088] Spatial scaling

[0089] (3) Same reference picture list, but different reference picture (different POC)

[0090] (4) Different reference picture list, and different reference picture (different POC)

[0091] The no-spatial-scaling cases are checked first followed by the spatial scaling. Spatial scaling is considered when the POC is different between the reference picture of the neighbouring PU and that of the current PU regardless of reference picture list. If all PUs of left candidates are not available or are intra coded, scaling for the above motion vector is allowed to help parallel derivation of left and above MV candidates. Otherwise, spatial scaling is not allowed for the above motion vector.

[0092] In a spatial scaling process, the motion vector of the neighbouring PU is scaled in a similar manner as for temporal scaling, as depicted as FIG. 9. The main difference is that the reference picture list and index of current PU is given as input; the actual scaling process is the

same as that of temporal scaling.

[0093] 2.1.3.3 Temporal motion vector candidates

[0094] Apart for the reference picture index derivation, all processes for the derivation of temporal merge candidates are the same as for the derivation of spatial motion vector candidates (see FIG. 6). The reference picture index is signalled to the decoder.

2.2 Local illumination compensation in JEM

[0095] Local Illumination Compensation (LIC) is based on a linear model for illumination changes, using a scaling factor a and an offset b . And it is enabled or disabled adaptively for each inter-mode coded coding unit (CU).

[0096] FIG. 10 shows neighbouring samples used for deriving IC parameters.

[0097] When LIC applies for a CU, a least square error method is employed to derive the parameters a and b by using the neighbouring samples of the current CU and their corresponding reference samples. More specifically, as illustrated in FIG. 12, the subsampled (2:1 subsampling) neighbouring samples of the CU and the corresponding samples (identified by motion information of the current CU or sub-CU) in the reference picture are used.

2.2.1 Derivation of prediction blocks

[0098] The LIC parameters are derived and applied for each prediction direction separately. For each prediction direction, a first prediction block is generated with the decoded motion information, then a temporary prediction block is obtained via applying the LIC model.

Afterwards, the two temporary prediction blocks are utilized to derive the final prediction block.

[0099] When a CU is coded with merge mode, the LIC flag is copied from neighbouring blocks, in a way similar to motion information copy in merge mode; otherwise, an LIC flag is signalled for the CU to indicate whether LIC applies or not.

[00100] When LIC is enabled for a picture, additional CU level RD check is needed to determine whether LIC is applied or not for a CU. When LIC is enabled for a CU, mean-removed sum of absolute difference (MR-SAD) and mean-removed sum of absolute Hadamard-transformed difference (MR-SATD) are used, instead of SAD and SATD, for integer pel motion refinement and fractional pel motion refinement, respectively.

[00101] To reduce the encoding complexity, the following encoding scheme is applied in the JEM.

- LIC is disabled for the entire picture when there is no obvious illumination change between a current picture and its reference pictures. To identify this situation, histograms of a current picture and every reference picture of the current picture are calculated at the encoder. If the histogram difference between the current picture and every reference picture of the current picture is smaller than a given threshold, LIC is disabled for the current picture; otherwise, LIC is enabled for the current picture.

2.3 Inter prediction methods in VVC

[00102] There are several new coding tools for inter prediction improvement, such as Adaptive motion vector difference resolution (AMVR) for signaling MVD, affine prediction mode, Triangular prediction mode (TPM), Advanced TMVP (ATMVP, aka SbTMVP), Generalized Bi-Prediction (GBI), Bi-directional Optical flow (BIO).

2.3.1 Coding block structure in VVC

[00103] In VVC, a QuadTree/BinaryTree/MultipleTree (QT/BT/TT) structure is adopted to divide a picture into square or rectangle blocks.

[00104] Besides QT/BT/TT, separate tree (a.k.a. Dual coding tree) is also adopted in VVC for I-frames. With separate tree, the coding block structure are signaled separately for the luma and chroma components.

2.3.2 Adaptive motion vector difference resolution

[00105] In HEVC, motion vector differences (MVDs) (between the motion vector and predicted motion vector of a PU) are signalled in units of quarter luma samples when `use_integer_mv_flag` is equal to 0 in the slice header. In the VVC, a locally adaptive motion vector resolution (AMVR) is introduced. In the VVC, MVD can be coded in units of quarter luma samples, integer luma samples or four luma samples (i.e., 1/4-pel, 1-pel, 4-pel). The MVD resolution is controlled at the coding unit (CU) level, and MVD resolution flags are conditionally signalled for each CU that has at least one non-zero MVD components.

[00106] For a CU that has at least one non-zero MVD components, a first flag is signalled to indicate whether quarter luma sample MV precision is used in the CU. When the first flag (equal to 1) indicates that quarter luma sample MV precision is not used, another flag is signalled to indicate whether integer luma sample MV precision or four luma sample MV precision is used.

[00107] When the first MVD resolution flag of a CU is zero, or not coded for a CU (meaning all MVDs in the CU are zero), the quarter luma sample MV resolution is used for the CU. When a CU uses integer-luma sample MV precision or four-luma-sample MV precision, the MVPs in

the AMVP candidate list for the CU are rounded to the corresponding precision.

2.3.3 Affine motion compensation prediction

[00108] 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 simplified affine transform motion compensation prediction is applied with 4-parameter affine model and 6-parameter affine model. As shown FIG. 13, the affine motion field of the block is described by two control point motion vectors (CPMVs) for the 4-parameter affine model and 3 CPMVs for the 6-parameter affine model.

[00109] FIG. 11 shows a simplified affine motion model for (a) 4-parameter affine (b) 6-parameter affine.

[00110] The motion vector field (MVF) of a block is described by the following equations with the 4-parameter affine model (wherein the 4-parameter are defined as the variables a , b , e and f) in equation (1) and 6-parameter affine model (wherein the 4-parameter are defined as the variables a , b , c , d , e and f) in equation (2) respectively:

$$\begin{cases} mv^h(x, y) = ax - by + e = \frac{(mv_1^h - mv_0^h)}{w}x - \frac{(mv_1^v - mv_0^v)}{w}y + mv_0^h \\ mv^v(x, y) = bx + ay + f = \frac{(mv_1^v - mv_0^v)}{w}x + \frac{(mv_1^h - mv_0^h)}{w}y + mv_0^v \end{cases} \quad (1)$$

$$\begin{cases} mv^h(x, y) = ax + cy + e = \frac{(mv_1^h - mv_0^h)}{w}x + \frac{(mv_2^h - mv_0^h)}{h}y + mv_0^h \\ mv^v(x, y) = bx + dy + f = \frac{(mv_1^v - mv_0^v)}{w}x + \frac{(mv_2^v - mv_0^v)}{h}y + mv_0^v \end{cases} \quad (2)$$

[00111] where (mv_0^h, mv_0^v) is motion vector of the top-left corner control point, and (mv_1^h, mv_1^v) is motion vector of the top-right corner control point and (mv_2^h, mv_2^v) is motion vector of the bottom-left corner control point, all of the three motion vectors are called **control point motion vectors (CPMV)**, (x, y) represents the coordinate of a **representative point** relative to the top-left sample within current block and $(mv^h(x, y), mv^v(x, y))$ is the motion vector derived for a sample located at (x, y) . The CP motion vectors may be signaled (like in the affine AMVP mode) or derived on-the-fly (like in the affine merge mode). w and h are the width and height of the current block. In practice, the division is implemented by right-shift with a rounding operation. In VTM, the **representative point** is defined to be the center position of a

sub-block, e.g., when the coordinate of the left-top corner of a sub-block relative to the top-left sample within current block is (x_s, y_s) , the coordinate of the representative point is defined to be (x_s+2, y_s+2) . For each sub-block (i.e., 4×4 in VTM), the representative point is utilized to derive the motion vector for the whole sub-block.

[00112] In order to further simplify the motion compensation prediction, sub-block based affine transform prediction is applied. To derive motion vector of each $M \times N$ (both M and N are set to 4 in current VVC) sub-block, the motion vector of the center sample of each sub-block, as shown in FIG. 14, is calculated according to Equation (1) and (2), and rounded to $1/16$ fraction accuracy. Then the motion compensation interpolation filters for $1/16$ -pel are applied to generate the prediction of each sub-block with derived motion vector. The interpolation filters for $1/16$ -pel are introduced by the affine mode.

[00113] After MCP, the high accuracy motion vector of each sub-block is rounded and saved as the same accuracy as the normal motion vector.

– Signaling of affine prediction

[00114] Similar to the translational motion model, there are also two modes for signaling the side information due affine prediction. They are AFFINE_INTER and AFFINE_MERGE modes.

– AF_INTER mode

[00115] For CUs with both width and height larger than 8, AF_INTER mode can be applied. An affine flag in CU level is signalled in the bitstream to indicate whether AF_INTER mode is used.

[00116] In this mode, for each reference picture list (List 0 or List 1), an affine AMVP candidate list is constructed with three types of affine motion predictors in the following order, wherein each candidate includes the estimated CPMVs of the current block. The differences of the best CPMVs found at the encoder side (such as $mv_0\ mv_1\ mv_2$ in FIG. 17) and the estimated CPMVs are signalled. In addition, the index of affine AMVP candidate from which the estimated CPMVs are derived is further signalled.

1) Inherited affine motion predictors

[00117] The checking order is similar to that of spatial MVPs in HEVC AMVP list construction. First, a left inherited affine motion predictor is derived from the first block in $\{A1, A0\}$ that is affine coded and has the same reference picture as in current block. Second, an above

inherited affine motion predictor is derived from the first block in $\{B1, B0, B2\}$ that is affine coded and has the same reference picture as in current block. The five blocks A1, A0, B1, B0, B2 are depicted in FIG. 16.

[00118] Once a neighboring block is found to be coded with affine mode, the CPMVs of the coding unit covering the neighboring block are used to derive predictors of CPMVs of current block. For example, if A1 is coded with non-affine mode and A0 is coded with 4-parameter affine mode, the left inherited affine MV predictor will be derived from A0. In this case, the CPMVs of a CU covering A0, as denoted by MV_0^N for the top-left CPMV and MV_1^N for the top-right CPMV in FIG. 18 (b) are utilized to derive the estimated CPMVs of current block, denoted by MV_0^C , MV_1^C , MV_2^C for the top-left (with coordinate (x0, y0)), top-right (with coordinate (x1, y1)) and bottom-right positions (with coordinate (x2, y2)) of current block.

2) Constructed affine motion predictors

[00119] A constructed affine motion predictor consists of control-point motion vectors (CPMVs) that are derived from neighboring inter coded blocks, as shown in FIG. 17, that have the same reference picture. If the current affine motion model is 4-parameter affine, the number of CPMVs is 2, otherwise if the current affine motion model is 6-parameter affine, the number of CPMVs is 3. The top-left CPMV $\overline{mv}_0$ is derived by the MV at the first block in the group $\{A, B, C\}$ that is inter coded and has the same reference picture as in current block. The top-right CPMV $\overline{mv}_1$ is derived by the MV at the first block in the group $\{D, E\}$ that is inter coded and has the same reference picture as in current block. The bottom-left CPMV $\overline{mv}_2$ is derived by the MV at the first block in the group $\{F, G\}$ that is inter coded and has the same reference picture as in current block.

[00120] If the current affine motion model is 4-parameter affine, then a constructed affine motion predictor is inserted into the candidate list only if both $\overline{mv}_0$ and $\overline{mv}_1$ are founded, that is, $\overline{mv}_0$ and $\overline{mv}_1$ are used as the estimated CPMVs for top-left (with coordinate (x0, y0)), top-right (with coordinate (x1, y1)) positions of current block.

[00121] If the current affine motion model is 6-parameter affine, then a constructed affine motion predictor is inserted into the candidate list only if $\overline{mv}_0$, $\overline{mv}_1$ and $\overline{mv}_2$ are all founded, that is, $\overline{mv}_0$, $\overline{mv}_1$ and $\overline{mv}_2$ are used as the estimated CPMVs for top-left (with coordinate (x0, y0)), top-right (with coordinate (x1, y1)) and bottom-right (with coordinate (x2, y2)) positions of current block.

[00122] No pruning process is applied when inserting a constructed affine motion predictor into the candidate list.

3) Normal AMVP motion predictors

[00123] The following applies until the number of affine motion predictors reaches the maximum.

- 1) Derive an affine motion predictor by setting all CPMVs equal to $\overline{mv}_2$ if available.
- 2) Derive an affine motion predictor by setting all CPMVs equal to $\overline{mv}_1$ if available.
- 3) Derive an affine motion predictor by setting all CPMVs equal to $\overline{mv}_0$ if available.
- 4) Derive an affine motion predictor by setting all CPMVs equal to HEVC TMVP if available.
- 5) Derive an affine motion predictor by setting all CPMVs to zero MV.

[00124] Note that $\overline{mv}_i$ is already derived in constructed affine motion predictor.

[00125] FIG. 14 shows MVP for AF_INTER for inherited affine candidates.

[00126] FIG. 15 shows MVP for AF_INTER for constructed affine candidates.

[00127] In AF_INTER mode, when 4/6-parameter affine mode is used, 2/3 control points are required, and therefore 2/3 MVD needs to be coded for these control points, as shown in FIG. 15. It is proposed to derive the MV as follows, i.e., mvd_1 and mvd_2 are predicted from mvd_0 .

$$mv_0 = \overline{mv}_0 + mvd_0$$

$$mv_1 = \overline{mv}_1 + mvd_1 + mvd_0$$

$$mv_2 = \overline{mv}_2 + mvd_2 + mvd_0$$

[00128] Wherein $\overline{mv}_i$, mvd_i and mv_i are the predicted motion vector, motion vector difference and motion vector of the top-left pixel ($i = 0$), top-right pixel ($i = 1$) or left-bottom pixel ($i = 2$) respectively, as shown in FIG. 15(b). Please note that the addition of two motion vectors (e.g., $mvA(xA, yA)$ and $mvB(xB, yB)$) is equal to summation of two components separately, that is, $newMV = mvA + mvB$ and the two components of newMV is set to $(xA + xB)$ and $(yA + yB)$, respectively.

– AF_MERGE mode

[00129] When a CU is applied in AF_MERGE mode, it gets the first block coded with affine mode from the valid neighbour reconstructed blocks. And the selection order for the candidate block is from left, above, above right, left bottom to above left as shown in FIG. 18(a) (denoted by A, B, C, D, E in order). For example, if the neighbour left bottom block is coded in affine

mode as denoted by A0 in FIG. 18(b), the Control Point (CP) motion vectors mv_0^N , mv_1^N and mv_2^N of the top left corner, above right corner and left bottom corner of the neighbouring CU/PU which contains the block A are fetched. And the motion vector mv_0^C , mv_1^C and mv_2^C (which is only used for the 6-parameter affine model) of the top left corner/top right/bottom left on the current CU/PU is calculated based on mv_0^N , mv_1^N and mv_2^N . It should be noted that in VTM-2.0, sub-block (e.g. 4×4 block in VTM) located at the top-left corner stores mv0, the sub-block located at the top-right corner stores mv1 if the current block is affine coded. If the current block is coded with the 6-parameter affine model, the sub-block located at the bottom-left corner stores mv2; otherwise (with the 4-parameter affine model), LB stores mv2'. Other sub-blocks stores the MVs used for MC.

[00130] After the CPMV of the current CU mv_0^C , mv_1^C and mv_2^C are derived, according to the simplified affine motion model Equation (1) and (2), the MVF of the current CU is generated. In order to identify whether the current CU is coded with AF_MERGE mode, an affine flag is signalled in the bitstream when there is at least one neighbour block is coded in affine mode.

[00131] An affine merge candidate list is constructed with following steps:

1) Insert inherited affine candidates

Inherited affine candidate means that the candidate is derived from the affine motion model of its valid neighbor affine coded block. The maximum two inherited affine candidates are derived from affine motion model of the neighboring blocks and inserted into the candidate list. For the left predictor, the scan order is {A0, A1}; for the above predictor, the scan order is {B0, B1, B2}.

2) Insert constructed affine candidates

If the number of candidates in affine merge candidate list is less than MaxNumAffineCand (e.g., 5), constructed affine candidates are inserted into the candidate list. Constructed affine candidate means the candidate is constructed by combining the neighbor motion information of each control point.

- a) The motion information for the control points is derived firstly from the specified spatial neighbors and temporal neighbor shown in FIG. 19. CP_k (k=1, 2, 3, 4) represents the k-th control point. A0, A1, A2, B0, B1, B2 and B3 are spatial positions for predicting CP_k (k=1, 2, 3); T is temporal position for predicting CP4.

The coordinates of CP1, CP2, CP3 and CP4 is (0, 0), (W, 0), (H, 0) and (W, H), respectively, where W and H are the width and height of current block.

[00132] The motion information of each control point is obtained according to the following priority order:

- For CP1, the checking priority is B2->B3->A2. B2 is used if it is available. Otherwise, if B2 is available, B3 is used. If both B2 and B3 are unavailable, A2 is used. If all the three candidates are unavailable, the motion information of CP1 cannot be obtained.
- For CP2, the checking priority is B1->B0.
- For CP3, the checking priority is A1->A0.
- For CP4, T is used.

b) Secondly, the combinations of controls points are used to construct an affine merge candidate.

I. Motion information of three control points are needed to construct a 6-parameter affine candidate. The three control points can be selected from one of the following four combinations ({CP1, CP2, CP4}, {CP1, CP2, CP3}, {CP2, CP3, CP4}, {CP1, CP3, CP4}). Combinations {CP1, CP2, CP3}, {CP2, CP3, CP4}, {CP1, CP3, CP4} will be converted to a 6-parameter motion model represented by top-left, top-right and bottom-left control points.

II. Motion information of two control points are needed to construct a 4-parameter affine candidate. The two control points can be selected from one of the two combinations ({CP1, CP2}, {CP1, CP3}). The two combinations will be converted to a 4-parameter motion model represented by top-left and top-right control points.

III. The combinations of constructed affine candidates are inserted into to candidate list as following order:

{CP1, CP2, CP3}, {CP1, CP2, CP4}, {CP1, CP3, CP4}, {CP2, CP3, CP4}, {CP1, CP2}, {CP1, CP3}

- i. For each combination, the reference indices of list X for each CP are checked, if they are all the same, then this combination has valid CPMVs for list X. If the combination does not have valid CPMVs for both list 0 and list 1, then this combination is marked as invalid. Otherwise, it is valid, and the CPMVs are put into the sub-block merge list.

3) Padding with zero motion vectors

[00133] If the number of candidates in affine merge candidate list is less than 5, zero motion vectors with zero reference indices are insert into the candidate list, until the list is full.

[00134] More specifically, for the sub-block merge candidate list, a 4-parameter merge candidate with MVs set to (0, 0) and prediction direction set to uni-prediction from list 0 (for P slice) and bi-prediction (for B slice).

[00135] In VTM4, the CPMVs of affine CUs are stored in a separate buffer. The stored CPMVs are only used to generate the inherited CPMVPs in affine merge mode and affine AMVP mode for the lately coded CUs. The sub-block MVs derived from CPMVs are used for motion compensation, MV derivation of merge/AMVP list of translational MVs and de-blocking. To avoid the picture line buffer for the additional CPMVs, affine motion data inheritance from the CUs from above CTU is treated differently to the inheritance from the normal neighboring CUs. If the candidate CU for affine motion data inheritance is in the above

CTU line, the bottom-left and bottom-right sub-block MVs in the line buffer instead of the CPMVs are used for the affine MVP derivation. In this way, the CPMVs are only stored in local buffer. If the candidate CU is 6-parameter affine coded, the affine model is degraded to 4-parameter model.

– **Merge with motion vector differences (MMVD)**

[00136] Ultimate motion vector expression (UMVE, also known as MMVD) is presented.

UMVE is used for either skip or merge modes with a proposed motion vector expression method.

[00137] UMVE re-uses merge candidate as same as those included in the regular merge candidate list in VVC. Among the merge candidates, a base candidate can be selected, and is further expanded by the proposed motion vector expression method.

[00138] UMVE provides a new motion vector difference (MVD) representation method, in which a starting point, a motion magnitude and a motion direction are used to represent a MVD.

[00139] This proposed technique uses a merge candidate list as it is. But only candidates which are default merge type (MRG_TYPE_DEFAULT_N) are considered for UMVE's expansion.

[00140] Base candidate index defines the starting point. Base candidate index indicates the best candidate among candidates in the list as follows.

Table 1. Base candidate IDX

Base candidate IDX	0	1	2	3
N th MVP	1 st MVP	2 nd MVP	3 rd MVP	4 th MVP

[00141] If the number of base candidate is equal to 1, Base candidate IDX is not signaled.

[00142] Distance index is motion magnitude information. Distance index indicates the pre-defined distance from the starting point information. Pre-defined distance is as follows:

Table 2. Distance IDX

Distance IDX	0	1	2	3	4	5	6	7
-----------------	---	---	---	---	---	---	---	---

Pixel distance	1/4-pel	1/2-pel	1-pel	2-pel	4-pel	8-pel	16-pel	32-pel
-----------------------	---------	---------	-------	-------	-------	-------	--------	--------

[00143] Direction index represents the direction of the MVD relative to the starting point. The direction index can represent of the four directions as shown below.

Table 3. Direction IDX

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

[00144] UMVE flag is signaled right after sending a skip flag or merge flag. If skip or merge flag is true, UMVE flag is parsed. If UMVE flag is equal to 1, UMVE syntaxes are parsed. But, if not 1, AFFINE flag is parsed. If AFFINE flag is equal to 1, that is AFFINE mode, But, if not 1, skip/merge index is parsed for VTM's skip/merge mode.

[00145] Additional line buffer due to UMVE candidates is not needed. Because a skip/merge candidate of software is directly used as a base candidate. Using input UMVE index, the supplement of MV is decided right before motion compensation. There is no need to hold long line buffer for this.

[00146] In current common test condition, either the first or the second merge candidate in the merge candidate list could be selected as the base candidate.

[00147] UMVE is also known as Merge with MV Differences (MMVD).

[00148] In addition, a flag, **tile_group_fpel_mmvd_enabled_flag**, is signalled to the decoder in the slice header whether to use fractional distances or not. When fractional distances are disabled, distances in the default table are all multiplied by 4, i.e., the distance table {1, 2, 4, 8, 16, 32, 64, 128}-pel is used. Because size of the distance table is not changed, entropy coding of the distance index is not changed.

– Decoder-side Motion Vector Refinement (DMVR)

[00149] In bi-prediction operation, for the prediction of one block region, two prediction blocks, formed using a motion vector (MV) of list0 and a MV of list1, respectively, are combined to form a single prediction signal. In the decoder-side motion vector refinement

(DMVR) method, the two motion vectors of the bi-prediction are further refined.

– DMVR in JEM

[00150] In JEM design, the motion vectors are refined by a bilateral template matching process. The bilateral template matching applied in the decoder to perform a distortion-based search between a bilateral template and the reconstruction samples in the reference pictures in order to obtain a refined MV without transmission of additional motion information. An example is depicted in FIG. 22. The bilateral template is generated as the weighted combination (i.e. average) of the two prediction blocks, from the initial MV0 of list0 and MV1 of list1, respectively, as shown in FIG. 22. The template matching operation consists of calculating cost measures between the generated template and the sample region (around the initial prediction block) in the reference picture. For each of the two reference pictures, the MV that yields the minimum template cost is considered as the updated MV of that list to replace the original one. In the JEM, nine MV candidates are searched for each list. The nine MV candidates include the original MV and 8 surrounding MVs with one luma sample offset to the original MV in either the horizontal or vertical direction, or both. Finally, the two new MVs, i.e., MV0' and MV1' as shown in FIG. 22, are used for generating the final bi-prediction results. A sum of absolute differences (SAD) is used as the cost measure. Please note that when calculating the cost of a prediction block generated by one surrounding MV, the rounded MV (to integer pel) is actually used to obtain the prediction block instead of the real MV.

[00151] FIG. 20 shows DMVR based on bilateral template matching.

– DMVR in VVC

[00152] For DMVR in VVC, MVD mirroring between list 0 and list 1 is assumed as shown in FIG. 21, and bilateral matching is performed to refine the MVs, i.e., to find the best MVD among several MVD candidates. Denote the MVs for two reference picture lists by $MVL0(L0X, L0Y)$, and $MVL1(L1X, L1Y)$. The MVD denoted by $(MvdX, MvdY)$ for list 0 that could minimize the cost function (e.g., SAD) is defined as the best MVD. For the SAD function, it is defined as the SAD between the reference block of list 0 derived with a motion vector $(L0X+MvdX, L0Y+MvdY)$ in the list 0 reference picture and the reference block of list 1 derived with a motion vector $(L1X-MvdX, L1Y-MvdY)$ in the list 1 reference picture.

[00153] The motion vector refinement process may iterate twice. In each iteration, at most 6 MVDs (with integer-pel precision) may be checked in two steps, as shown in FIG. 22. In the first step, MVD (0, 0), (-1, 0), (1, 0), (0, -1), (0, 1) are checked. In the second step, one of the MVD (-1, -1), (-1, 1), (1, -1) or (1, 1) may be selected and further checked. Suppose function $Sad(x, y)$ returns SAD value of the MVD (x, y). The MVD, denoted by (MvdX, MvdY), checked in the second step is decided as follows:

```

MvdX = -1;
MvdY = -1;
If (Sad(1, 0) < Sad(-1, 0))
    MvdX = 1;
If (Sad(0, 1) < Sad(0, -1))
    MvdY = 1;

```

[00154] In the first iteration, the starting point is the signaled MV, and in the second iteration, the starting point is the signaled MV plus the selected best MVD in the first iteration. DMVR applies only when one reference picture is a preceding picture and the other reference picture is a following picture, and the two reference pictures are with same picture order count distance from the current picture.

[00155] FIG. 21 is an example of MVD (0, 1) mirrored between list 0 and list 1 in DMVR.

[00156] FIG. 22 shows MVs that may be checked in one iteration.

Also, the DMVR in VVC first performs integer MVD refinement as described above. This is the first step. After that, an MVD refinement on fractional precision is conditionally performed to further refine the motion vector. This is the second step. The condition whether to perform the second step is based on whether the MVD after the current iteration is a zero MV. If it's a zero MV (vertical and horizontal components of a MV is 0.), the second step will be performed.

[00157] The details of fractional MVD refinement are given as follows. It should be noted that the MVD denotes the different between the initial motion vector and the final motion vector used in motion compensation.

[00158] A parametric error surface is fitted using integer distance positions and the costs evaluated in these positions, which is then used to determine 1/16th-pel precision sub-pixel offsets.

[00159] The proposed method is summarized below:

1. The parametric error surface fit is computed only if the center position is the best cost position in a given iteration.
2. The center position cost and the costs at (-1,0), (0,-1), (1,0) and (0,1) positions from the center are used to fit a 2-D parabolic error surface equation of the form

$$E(x, y) = A(x - x_0)^2 + B(y - y_0)^2 + C$$

where (x_0, y_0) corresponds to the position with the least cost and C corresponds to the minimum cost value. By solving the 5 equations in 5 unknowns, (x_0, y_0) is computed as:

$$x_0 = (E(-1,0) - E(1,0)) / (2(E(-1,0) + E(1,0) - 2E(0,0)))$$

$$y_0 = (E(0,-1) - E(0,1)) / (2(E(0,-1) + E(0,1) - 2E(0,0)))$$

(x_0, y_0) can be computed to any required sub-pixel precision by adjusting the precision at which the division is performed (i.e. how many bits of quotient are computed). For 1/16th-pel accuracy, just 4-bits in the absolute value of the quotient needs to be computed, which lends itself to a fast shifted subtraction based implementation of the 2 divisions required per CU.

3. The computed (x_0, y_0) are added to the integer distance refinement MV to get the sub-pixel accurate refinement delta MV.

Magnitude of the derived fraction motion vector is constrained to be smaller than or equal to half pixel.

[00160] To further simplify the process of DMVR, it proposed several changes to the design in JEM. More specifically, the adopted DMVR design to VTM-4.0 (to be released soon) has the following main features:

- Early termination when (0,0) position SAD between list0 and list1 is smaller than a threshold.
- Early termination when SAD between list0 and list1 is zero for some position.
- Block sizes for DMVR: $W \cdot H \geq 64$ & $H \geq 8$, wherein W and H are the width and height of the block.
- Split the CU into multiple of 16x16 sub-blocks for DMVR of CU size $> 16 \cdot 16$. If only width or height of the CU is larger than 16, it is only split in vertical or horizontal direction.
- Reference block size $(W+7) \cdot (H+7)$ (for luma).
- 25 points SAD-based integer-pel search (i.e. $(+/-)$ 2 refinement search range, single stage)
- Bilinear-interpolation based DMVR.
- “Parametric error surface equation” based sub-pel refinement. This procedure is performed only when the minimum SAD cost is not equal to zero and the best MVD is (0, 0) in the last MV refinement iteration.
- Luma/chroma MC w/ reference block padding (if needed).
- Refined MVs used for MC and TMVPs only.

– **Usage of DMVR**

[00161] When the following conditions are all true, DMVR may be enabled:

- DMVR enabling flag in the SPS (i.e., `sps_dmvr_enabled_flag`) is equal to 1
- TPM flag, inter-affine flag and subblock merge flag (either ATMVP or affine merge), MMVD flag are all equal to 0
- Merge flag is equal to 1
- Current block is bi-predicted, and POC distance between current picture and reference picture in list 1 is equal to the POC distance between reference picture in list 0 and current picture
- The current CU height is greater than or equal to 8
- Number of luma samples (CU width*height) is greater than or equal to 64

– **Required reference samples in DMVR**

[00162] For a block with size $W * H$, suppose the maximum allowable MVD value is +/- offset (for example 2 in VVC), and the filter size is `filterSize` (for example, 8 for luma and 4 for chroma in VVC), then $(W + 2 * \text{offset} + \text{filterSize} - 1) * (H + 2 * \text{offset} + \text{filterSize} - 1)$ reference samples are required. To reduce the memory bandwidth, only the center $(W + \text{filterSize} - 1) * (H + \text{filterSize} - 1)$ reference samples are fetched, and the other pixels are generated by repeating the boundary of the fetched samples. Example for 8x8 block is shown in FIG. 23.

[00163] During the motion vector refinement, bilinear motion compensation is performed using these reference samples. Meanwhile, the final motion compensation is also performed using these reference samples.

[00164] FIG. 23 shows an example of required reference samples with padding.

– **Combined intra and inter prediction(CIIP)**

[00165] Multi-hypothesis prediction is proposed, wherein combined intra and inter prediction is one way to generate multiple hypotheses.

[00166] When the multi-hypothesis prediction is applied to improve intra mode, multi-hypothesis prediction combines one intra prediction and one merge indexed prediction. In a merge CU, one flag is signaled for merge mode to select an intra mode from an intra candidate list when the flag is true. For luma component, the intra candidate list is derived from 4 intra prediction modes including DC, planar, horizontal, and vertical modes, and the size of the intra candidate list can be 3 or 4 depending on the block shape. When the CU width is larger than the double of CU height, horizontal mode is exclusive of the intra mode list and when the CU height

is larger than the double of CU width, vertical mode is removed from the intra mode list. One intra prediction mode selected by the intra mode index and one merge indexed prediction selected by the merge index are combined using weighted average. For chroma component, DM is always applied without extra signaling. The weights for combining predictions are described as follow. When DC or planar mode is selected, or the CB width or height is smaller than 4, equal weights are applied. For those CBs with CB width and height larger than or equal to 4, when horizontal/vertical mode is selected, one CB is first vertically/horizontally split into four equal-area regions. Each weight set, denoted as (w_intra_i, w_inter_i) , where i is from 1 to 4 and $(w_intra_1, w_inter_1) = (6, 2)$, $(w_intra_2, w_inter_2) = (5, 3)$, $(w_intra_3, w_inter_3) = (3, 5)$, and $(w_intra_4, w_inter_4) = (2, 6)$, will be applied to a corresponding region. (w_intra_1, w_inter_1) is for the region closest to the reference samples and (w_intra_4, w_inter_4) is for the region farthest away from the reference samples. Then, the combined prediction can be calculated by summing up the two weighted predictions and right-shifting 3 bits. Moreover, the intra prediction mode for the intra hypothesis of predictors can be saved for reference of the following neighboring CUs.

[00167] In VTM4, when a CU is coded in merge mode, and if the CU contains at least 64 luma samples (that is, CU width times CU height is equal to or larger than 64), an additional flag is signalled to indicate if the combined inter/intra prediction (CIIP) mode is applied to the current CU.

3. Problems

[00168] CIIP employs a weighted combination between intra and inter prediction, which may be not efficient on coding screen contents with sharp edges. It may blur the prediction signal then further comprise the coding performance.

4. Example listing of embodiments and techniques

[00169] It is proposed that certain samples within one block may be only predicted from intra prediction while other may be only predicted from inter prediction when CIIP is enabled for this block.

[00170] The detailed listing below should be considered as examples to explain general concepts. These techniques should not be interpreted in a narrow way. Furthermore, these inventions can be combined in any manner.

[00171] The methods described below may be also applicable to other decoder motion

information derivation technologies in addition to the CIIP mentioned below.

1. The weight may be updated in the CIIP mode for a prediction unit/coding block/region
 - a. In one example, the weight sets in CIIP mode, denotes as (w_intra_i, w_inter_i) , where i is from 1 to 4, may be updated to $(w_intra_1, w_inter_1) = (N, 0)$, $(w_intra_2, w_inter_2) = (0, N)$, $(w_intra_3, w_inter_3) = (0, N)$, and $(w_intra_4, w_inter_4) = (0, N)$
 - b. In one example, the weight sets in CIIP mode, denotes as (w_intra_i, w_inter_i) , where i is from 1 to 4, may be updated to $(w_intra_1, w_inter_1) = (N, 0)$, $(w_intra_2, w_inter_2) = (N, 0)$, $(w_intra_3, w_inter_3) = (0, N)$, and $(w_intra_4, w_inter_4) = (0, N)$
 - c. In one example, the weight sets in CIIP mode, denotes as (w_intra_i, w_inter_i) , where i is from 1 to 4, may be updated to $(w_intra_1, w_inter_1) = (N, 0)$, $(w_intra_2, w_inter_2) = (N, 0)$, $(w_intra_3, w_inter_3) = (N, 0)$, and $(w_intra_4, w_inter_4) = (0, N)$
 - d. In one example, weights of only some parts are updated according to above methods and other parts still use the current weights.
 - e. In one example, the upper k rows' of pixels may have (intra, inter) weights $(N, 0)$, and the other rows may have weights $(0, N)$.
 - f. In one example, the left k columns' of pixels may have (intra, inter) weights $(N, 0)$, and the other columns may have weights $(0, N)$.
 - g. In one example, N is set to 1. Alternatively, N is set to 8 when the final prediction block is the average of two prediction blocks divided by 8.
 - h. In one example, the weighted prediction can be implemented as block copy in CIIP mode when using the above updated weight sets
 - i. In one example, the value of updated weights in CIIP mode for a prediction unit/coding block/region may depend on a message (e.g. a flag) signaled such as in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU).
2. The indicate of usage of updating weight in CIIP mode for a prediction unit/coding block/region may depend on a message (e.g. a flag) signaled such as in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU).
 - a. In one example, a flag, such as *tile_group_MHIntra_SCC_weight_enabled_flag*, signaled in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU) is true, the weight in CIIP mode may be updated. Alternative, the weight in CIIP mode may be updated when a flag signaled in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU) is true
 - b. In one example, a flag, such as *tile_group_MHIntra_SCC_weight_enabled_flag*, signaled in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU) is false, the weight in CIIP mode may be not updated. Alternative, the weight in CIIP mode may be not updated when a flag signaled in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU) is false
 - c. In one example, the indication of usage of updating weights in the CIIP mode may be inferred and may depend on

- a) Current block dimension
 - b) Current quantization parameter
 - c) Transform type
 - d) Coding mode of the block that Inter motion vector points to
 - e) Motion vector precision
 - f) The merge index used in CIIP
 - g) The intra prediction mode used in CIIP
 - h) The magnitudes of motion vectors used in CIIP
 - i) The indication of which weight set used by the neighboring blocks
 - d. In one example, the indication of usage of updating the weight in the CIIP mode may be signaled at sub-block level
 - e. In one example, the indication of usage of updating the weight in the CIIP mode may be inferred at sub-block level
3. Multiple weight set may be provided in CIIP mode. The indicate of usage of which weight set for a prediction unit/coding block/region may depend on a message (e.g. a flag) signaled such as in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU).
- a. In one example, a message (e.g. a flag) may be signaled in the sequence (e.g. SPS)/slice (e.g. slice header)/tile group (e.g. tile group header)/picture level (e.g. picture header)/block level (e.g. CTU or CU). to indicate which weight set may be used in the CIIP mode.
 - b. In one example, the indication of which weight set used in the CIIP mode may be inferred, alternatively, furthermore, in one example, which may be inferred based on
 - a) Current block dimension
 - b) Current quantization parameter
 - c) The merge index used in CIIP
 - d) The intra prediction mode used in CIIP
 - e) The magnitudes of motion vectors used in CIIP
 - f) The indication of which weight set used by the neighboring blocks
 - c. In one example, the indication of which weight set used in the CIIP mode may be signaled at sub-block level
 - d. In one example, the indication of which weight set used in the CIIP mode may be inferred at sub-block level

– Embodiment

7.3.2 Raw byte sequence payloads, trailing bits and byte alignment syntax

7.3.2.1 Sequence parameter set RBSP syntax

seq_parameter_set_rbsp() {	Descriptor
sps_max_sub_layers_minus1	u(3)
sps_reserved_zero_5bits	u(5)
...	
sps_sbt_enabled_flag	u(1)
if(sps_sbt_enabled_flag)	
sps_sbt_max_size_64_flag	u(1)
sps_affine_enabled_flag	u(1)
if(sps_affine_enabled_flag)	
sps_affine_type_flag	u(1)

sps_gbi_enabled_flag	u(1)
sps_ibc_enabled_flag	u(1)
sps_ciip_enabled_flag	u(1)
<i>if(sps_ciip_enabled_flag)</i>	
sps_scc_MHIntra_weight_enabled_flag	u(1)
...	
}	

sps_scc_MHIntra_weight_enabled_flag specifies that the weight in the CIIP mode is updated. **sps_scc_MHIntra_weight_enabled_flag** equal to 0 specifies that the weight in the CIIP mode is not updated. If the **sps_scc_MHIntra_weight_enabled_flag** is not present, it is inferred to 0.

7.3.4 Tile group header syntax

7.3.4.1 General tile group header syntax

tile_group_header() {	Descriptor
tile_group_pic_parameter_set_id	ue(v)
...	
if((weighted_pred_flag && tile_group_type == P) (weighted_bipred_flag && tile_group == B))	
pred_weight_table()	
six_minus_max_num_merge_cand	ue(v)
if(sps_affine_enabled_flag)	
five_minus_max_num_subblock_merge_cand	ue(v)
if(sps_fpel_mmvd_enabled_flag)	
tile_group_fpel_mmvd_enabled_flag	u(1)
} else if (sps_ibc_enabled_flag)	
six_minus_max_num_merge_cand	ue(v)
<i>if(sps_scc_MHIntra_weight_enabled_flag)</i>	
tile_group_MHIntra_SCC_weight_enabled_flag	u(1)
tile_group_qp_delta	se(v)
if(pps_tile_group_chroma_qp_offsets_present_flag) {	
tile_group_cb_qp_offset	se(v)
tile_group_cr_qp_offset	se(v)
...	
}	

8.5.7.6 Weighted sample prediction process for combined merge and intra prediction

Inputs to this process are:

- the width of the current coding block cbWidth,
- the height of the current coding block cbHeight,
- two (cbWidth)x(cbHeight) arrays predSamplesInter and predSamplesIntra,
- the intra prediction mode predModeIntra,

- a variable cIdx specifying the colour component index.

Output of this process is the (cbWidth)x(cbHeight) array predSamplesComb of prediction sample values.

The variable bitDepth is derived as follows:

- If cIdx is equal to 0, bitDepth is set equal to BitDepth_Y.
- Otherwise, bitDepth is set equal to BitDepth_C.

The prediction samples predSamplesComb[x][y] with x = 0..cbWidth – 1 and y = 0..cbHeight – 1 are derived as follows:

- The weight w is derived as follows:
 - If one or more of the following conditions are true, w is set equal to 4:
 - cbWidth is less than 4.
 - cbHeight is less than 4.
 - predModeIntra is equal to INTRA_PLANAR
 - predModeIntra is equal to INTRA_DC.
 - Otherwise, if predModeIntra is INTRA_ANGULAR50, w is specified in Table 8-11 with nPos equal to y and nSize equal to cbHeight.
 - Otherwise, if predModeIntra is INTRA_ANGULAR18, w is specified in Table 8-11 with nPos equal to x and nSize equal to cbWidth.
 - Otherwise, w is set equal to 4.
- The prediction samples predSamplesComb[x][y] are derived as follows:

$$\begin{aligned} &\text{-if tile_group_MHIntra_SCC_weight_enabled_flag is 1,} \\ &\quad \text{predSamplesComb[x][y]} = (w1 * \text{predSamplesIntra[x][y]} + \\ &\quad (8 - w1) * \text{predSamplesInter[x][y]}) >> 3 \end{aligned} \quad (8-888)$$

$$\begin{aligned} &\text{-if tile_group_MHIntra_SCC_weight_enabled_flag is 0,} \\ &\quad \text{predSamplesComb[x][y]} = (w2 * \text{predSamplesIntra[x][y]} + \\ &\quad (8 - w2) * \text{predSamplesInter[x][y]}) >> 3 \end{aligned} \quad (8-889)$$

Table 8-11 – Specification of w1 as a function of the position nP and the size nS

$0 \leq nP < (nS / 4)$	$(nS / 4) \leq nP < (nS / 2)$	$(nS / 2) \leq nP < (3 * nS / 4)$	$(3 * nS / 4) \leq nP < nS$
8	0	0	0

Table 8-12 – Specification of w2 as a function of the position nP and the size nS

$0 \leq nP < (nS / 4)$	$(nS / 4) \leq nP < (nS / 2)$	$(nS / 2) \leq nP < (3 * nS / 4)$	$(3 * nS / 4) \leq nP < nS$
6	5	3	2

[00172] FIG. 24 is a block diagram of a video processing apparatus 1000. The apparatus 1000 may be used to implement one or more of the methods described herein. The apparatus 1000 may be embodied in a smartphone, tablet, computer, Internet of Things (IoT) receiver, and so on. The apparatus 1000 may include one or more processors 1002, one or more memories 1004 and video processing hardware 1006. The processor(s) 1002 may be configured to implement one or more methods described in the present document. The memory (memories) 1004 may be used for storing data and code used for implementing the methods and techniques described herein. The video processing hardware 1006 may be used to implement, in hardware circuitry, some techniques described in the present document.

[00173] FIG. 25 is a flowchart for an example method 2500 of video processing. The method 2500 includes updating (2502), during a conversion between a current video region and a bitstream representation of the current video block, weights used for a combined inter and intra prediction mode and performing (2504) the conversion based on the updating.

[00174] Additional features are described in CLAIMS section and Section 4.

[00175] FIG. 26 is a flowchart for an example method 2600 of video processing. The method 2600 includes determining (2502), for a conversion between a first block of video and a bitstream representation of the first block of video, whether usage of updating weight in a Combined intra and inter prediction (CIIP) mode to be applied during the conversion is enabled or disabled; updating (2504) the weights applied in partial pixels of the first block in CIIP mode in response to the determination that updating weight in CIIP mode is enabled; and performing (2506) the conversion based on the updated weights and non-updated weights.

[00176] In some examples, the CIIP mode is applied to derive a final prediction of the first block based on a weighted sum of an intermediate intra prediction and an intermediate merge inter prediction of the first block.

[00177] In some examples, the CIIP mode is applied for at least one of a prediction unit, coding block and region during the conversion.

[00178] In some examples, the weights applied in partial pixels of the first block in the CIIP mode include one or more weight sets, and each weight set (w_{intra_i} , w_{inter_i}) includes a weight for intra mode (w_{intra_i}) and a weight for inter mode (w_{inter_i}) and are applied to a corresponding prediction unit or coding block or region, where i is an integer.

[00179] In some examples, i is from 1 to 4.

[00180] In some examples, the weight set ($w_{\text{intra}1}$, $w_{\text{inter}1}$) is for a region closest to reference samples, and ($w_{\text{intra}4}$, $w_{\text{inter}4}$) is for a region farthest away from the reference samples.

[00181] In some examples, the weight sets in the CIIP mode are updated to ($w_{\text{intra}1}$, $w_{\text{inter}1}$) = (N , 0), ($w_{\text{intra}2}$, $w_{\text{inter}2}$) = (0 , N), ($w_{\text{intra}3}$, $w_{\text{inter}3}$) = (0 , N), and ($w_{\text{intra}4}$, $w_{\text{inter}4}$) = (0 , N), N being an integer.

[00182] In some examples, the weight sets in the CIIP mode are updated to ($w_{\text{intra}1}$, $w_{\text{inter}1}$) = (N , 0), ($w_{\text{intra}2}$, $w_{\text{inter}2}$) = (N , 0), ($w_{\text{intra}3}$, $w_{\text{inter}3}$) = (0 , N), and ($w_{\text{intra}4}$, $w_{\text{inter}4}$) = (0 , N), N being an integer.

[00183] In some examples, the weight sets in the CIIP mode are updated to ($w_{\text{intra}1}$, $w_{\text{inter}1}$) = (N , 0), ($w_{\text{intra}2}$, $w_{\text{inter}2}$) = (N , 0), ($w_{\text{intra}3}$, $w_{\text{inter}3}$) = (N , 0), and ($w_{\text{intra}4}$, $w_{\text{inter}4}$) = (0 , N), N being an integer.

[00184] In some examples, a weight set (w_{intra} , w_{inter}) for the pixels at the upper k rows of the first block has weights (N , 0), and a weight set (w_{intra} , w_{inter}) for the other rows has weights (0 , N), K being an integer.

[00185] In some examples, a weight set (w_{intra} , w_{inter}) for the pixels at the left k columns of the first block has weights (N , 0), and a weight set (w_{intra} , w_{inter}) for the other columns has weights (0 , N), K being an integer.

[00186] In some examples, N is set to 1.

[00187] In some examples, when a final prediction block is the weighted average of two prediction blocks divided by 8, N is set to 8.

[00188] In some examples, when the updated weight sets are used during the conversion, the weighted prediction is implemented as block copy in CIIP mode.

[00189] In some examples, the value of the weights applied in partial pixels of the first block in CIIP mode depends on a message signaled in at least one of the following: sequence level including sequence parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture header, and block level including coding tree unit (CTU) and coding unit (CU).

[00190] In some examples, the determination is based on an indication of the usage of updating weight in CIIP mode.

[00191] In some examples, the indication of usage of updating weight in CIIP mode depends on a message signaled in at least one of the following: sequence level including sequence

parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture header, and block level including coding tree unit (CTU) and coding unit (CU).

[00192] In some examples, the message includes a flag present in the bitstream representation.

[00193] In some examples, when the flag is true, the usage of updating weight in CIIP mode is enabled.

[00194] In some examples, when the flag is false, the usage of updating weight in CIIP mode is disabled.

[00195] In some examples, the flag is `tile_group_MHIIntra_SCC_weight_enabled_flag`.

[00196] In some examples, the indication of usage of updating weights in CIIP mode is inferred.

[00197] In some examples, the indication of usage of updating weights in the CIIP mode depends on at least one of :

- a) current block dimension;
- b) current quantization parameter;
- c) transform type;
- d) coding mode of the block that Inter motion vector points to;
- e) motion vector precision;
- f) the merge index used in CIIP mode;
- g) the intra prediction mode used in CIIP mode;
- h) the magnitudes of motion vectors used in CIIP mode;
- i) the indication of which weight set used by the neighboring blocks of the first block.

[00198] In some examples, the indication of usage of updating weights in CIIP mode is signaled at sub-block level.

[00199] In some examples, the indication of usage of updating weights in CIIP mode is inferred at sub-block level.

[00200] FIG. 27 is a flowchart for an example method 2700 of video processing. The method 2700 includes determining (2702), for a conversion between a first block of video and a bitstream representation of the first block of video, a weight set from multiple weight sets being used in a Combined intra and inter prediction (CIIP) mode depends on a message presented in at least one of the following: sequence level including sequence parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture

header, and block level including coding tree unit (CTU) and coding unit (CU); applying (2704) CIIP mode to generate a final prediction of the first block based on the determined weight set; and performing (2706) the conversion based on the final prediction; wherein the final prediction of the first block is generated based on a weighted sum of an intermediate intra prediction and an intermediate merge inter prediction of the first block.

[00201] In some examples, at least one of the multiple weight sets in the CIIP mode is applied for at least one of a prediction unit, coding block and region during the conversion.

[00202] In some examples, the message includes a flag present in the bitstream representation.

[00203] In some examples, the message is signaled to indicate which weight set is to be used in CIIP mode.

[00204] In some examples, an indication of which weight set of the multiple weight sets being used in CIIP mode is inferred.

[00205] In some examples, the indication of which weight set of the multiple weight sets being used in CIIP mode depends on at least one of :

- a) current block dimension;
- b) current quantization parameter;
- c) the merge index used in CIIP mode;
- d) the intra prediction mode used in CIIP mode;
- e) the magnitudes of motion vectors used in CIIP mode;
- f) the indication of which weight set used by the neighboring blocks of the first block.

[00206] In some examples, an indication of which weight set of the multiple weight sets being used in CIIP mode is signaled at sub-block level.

[00207] In some examples, an indication of which weight set of the multiple weight sets being used in CIIP mode is inferred at sub-block level.

[00208] In some examples, the conversion generates the first block of video from the bitstream representation.

[00209] In some examples, the conversion generates the bitstream representation from the first block of video.

[00210] The disclosed and other solutions, examples, embodiments, modules and the functional operations described in this document can be implemented in digital electronic circuitry, or in computer software, firmware, or hardware, including the structures disclosed in

this document and their structural equivalents, or in combinations of one or more of them. The disclosed and other embodiments can be implemented as one or more computer program products, i.e., one or more modules of computer program instructions encoded on a computer readable medium for execution by, or to control the operation of, data processing apparatus. The computer readable medium can be a machine-readable storage device, a machine-readable storage substrate, a memory device, a composition of matter effecting a machine-readable propagated signal, or a combination of one or more them. The term “data processing apparatus” encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them. A propagated signal is an artificially generated signal, e.g., a machine-generated electrical, optical, or electromagnetic signal, that is generated to encode information for transmission to suitable receiver apparatus.

[00211] A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[00212] The processes and logic flows described in this document can be performed by one or more programmable processors executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated

circuit).

[00213] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random-access memory or both. The essential elements of a computer are a processor for performing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Computer readable media suitable for storing computer program instructions and data include all forms of non-volatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[00214] While this patent document contains many specifics, these should not be construed as limitations on the scope of any subject matter or of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments of particular techniques. Certain features that are described in this patent document in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

[00215] Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. Moreover, the separation of various system components in the embodiments described in this patent document should not be understood as requiring such separation in all

embodiments.

[00216] Only a few implementations and examples are described and other implementations, enhancements and variations can be made based on what is described and illustrated in this patent document.

CLAIMS

1. A method of video processing, comprising:

determining, for a conversion between a first block of video and a bitstream representation of the first block of video, whether usage of updating weight in a Combined intra and inter prediction (CIIP) mode to be applied during the conversion is enabled or disabled;

updating the weights applied in partial pixels of the first block in CIIP mode in response to the determination that updating weight in CIIP mode is enabled; and

performing the conversion based on the updated weights and non-updated weights.

2. The method of claim 1, wherein the CIIP mode is applied to derive a final prediction of the first block based on a weighted sum of an intermediate intra prediction and an intermediate merge inter prediction of the first block.

3. The method of claim 1 or 2, wherein the CIIP mode is applied for at least one of a prediction unit, coding block and region during the conversion.

4. The method of any of claims 1-3, wherein the weights applied in partial pixels of the first block in the CIIP mode include one or more weight sets, and each weight set (w_intra_i, w_inter_i) includes a weight for intra mode (w_intra_i) and a weight for inter mode (w_inter_i) and are applied to a corresponding prediction unit or coding block or region, where i is an integer.

5. The method of claim 4, wherein i is from 1 to 4.

6. The method of claim 5, wherein the weight set (w_intra_1, w_inter_1) is for a region closest to reference samples, and (w_intra_4, w_inter_4) is for a region farthest away from the reference samples.

7. The method of claim 5 or 6, wherein the weight sets in the CIIP mode are updated to $(w_intra_1, w_inter_1) = (N, 0)$, $(w_intra_2, w_inter_2) = (0, N)$, $(w_intra_3, w_inter_3) = (0, N)$, and $(w_intra_4, w_inter_4) = (0, N)$, N being an integer.

8. The method of claim 5 or 6, wherein the weight sets in the CIIP mode are updated to $(w_intra_1, w_inter_1) = (N, 0)$, $(w_intra_2, w_inter_2) = (N, 0)$, $(w_intra_3, w_inter_3) = (0, N)$, and $(w_intra_4, w_inter_4) = (0, N)$, N being an integer.
9. The method of claim 5 or 6, wherein the weight sets in the CIIP mode are updated to $(w_intra_1, w_inter_1) = (N, 0)$, $(w_intra_2, w_inter_2) = (N, 0)$, $(w_intra_3, w_inter_3) = (N, 0)$, and $(w_intra_4, w_inter_4) = (0, N)$, N being an integer.
10. The method of any of claim 1-5, wherein a weight set (w_intra, w_inter) for the pixels at the upper k rows of the first block has weights $(N, 0)$, and a weight set (w_intra, w_inter) for the other rows has weights $(0, N)$, K being an integer.
11. The method of any of claim 1-5, wherein a weight set (w_intra, w_inter) for the pixels at the left k columns of the first block has weights $(N, 0)$, and a weight set (w_intra, w_inter) for the other columns has weights $(0, N)$, K being an integer.
12. The method of any of claims 6-11, wherein N is set to 1.
13. The method of any of claims 6-11, wherein when a final prediction block is the weighted average of two prediction blocks divided by 8, N is set to 8.
14. The method of any of claims 1-13, wherein when the updated weight sets are used during the conversion, the weighted prediction is implemented as block copy in CIIP mode.
15. The method of any of claims 1-14, wherein the value of the weights applied in partial pixels of the first block in CIIP mode depends on a message signaled in at least one of the following: sequence level including sequence parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture header, and block level including coding tree unit (CTU) and coding unit (CU).

16. The method of any of claims 1-15, wherein the determination is based on an indication of the usage of updating weight in CIIP mode.

17. The method of claim 16, wherein the indication of usage of updating weight in CIIP mode depends on a message signaled in at least one of the following: sequence level including sequence parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture header, and block level including coding tree unit (CTU) and coding unit (CU).

18. The method of claim 17, wherein the message includes a flag present in the bitstream representation.

19. The method of claim 18, wherein when the flag is true, the usage of updating weight in CIIP mode is enabled.

20. The method of claim 18, wherein when the flag is false, the usage of updating weight in CIIP mode is disabled.

21. The method of any of claims 18-20, wherein the flag is *tile_group_MHIntra_SCC_weight_enabled_flag*.

22. The method of claim 16, wherein the indication of usage of updating weights in CIIP mode is inferred.

23. The method of claim 22, wherein the indication of usage of updating weights in the CIIP mode depends on at least one of :

- a) current block dimension;
- b) current quantization parameter;
- c) transform type;
- d) coding mode of the block that Inter motion vector points to;
- e) motion vector precision;

- f) the merge index used in CIIP mode;
- g) the intra prediction mode used in CIIP mode;
- h) the magnitudes of motion vectors used in CIIP mode;
- i) the indication of which weight set used by the neighboring blocks of the first block.

24. The method of claim 16, wherein the indication of usage of updating weights in CIIP mode is signaled at sub-block level.

25. The method of claim 16, wherein the indication of usage of updating weights in CIIP mode is inferred at sub-block level.

26. A method of video processing, comprising:

determining, for a conversion between a first block of video and a bitstream representation of the first block of video, a weight set from multiple weight sets being used in a Combined intra and inter prediction (CIIP) mode depends on a message presented in at least one of the following: sequence level including sequence parameter set (SPS), slice level including slice header, tile group level including tile group header, picture level including picture header, and block level including coding tree unit (CTU) and coding unit (CU);

applying CIIP mode to generate a final prediction of the first block based on the determined weight set; and

performing the conversion based on the final prediction;

wherein the final prediction of the first block is generated based on a weighted sum of an intermediate intra prediction and an intermediate merge inter prediction of the first block.

27. The method of claim 26, wherein at least one of the multiple weight sets in the CIIP mode is applied for at least one of a prediction unit, coding block and region during the conversion.

28. The method of claim 26 or 27, wherein the message includes a flag present in the bitstream representation.

29. The method of any of claims 26-28, wherein the message is signaled to indicate which weight set is to be used in CIIP mode.
30. The method of claim 26 or 27, wherein an indication of which weight set of the multiple weight sets being used in CIIP mode is inferred.
31. The method of claim 30, wherein the indication of which weight set of the multiple weight sets being used in CIIP mode depends on at least one of :
- a) current block dimension;
 - b) current quantization parameter;
 - c) the merge index used in CIIP mode;
 - d) the intra prediction mode used in CIIP mode;
 - e) the magnitudes of motion vectors used in CIIP mode;
 - f) the indication of which weight set used by the neighboring blocks of the first block.
32. The method of claim 26 or 27, wherein an indication of which weight set of the multiple weight sets being used in CIIP mode is signaled at sub-block level.
33. The method of claim 26 or 27, wherein an indication of which weight set of the multiple weight sets being used in CIIP mode is inferred at sub-block level.
34. The method of anyone of claims 1-33, wherein the conversion generates the first block of video from the bitstream representation.
35. The method of anyone of claims 1-33, wherein the conversion generates the bitstream representation from the first block of video.
36. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of claims 1 to 35.

37. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of claims 1 to 35.

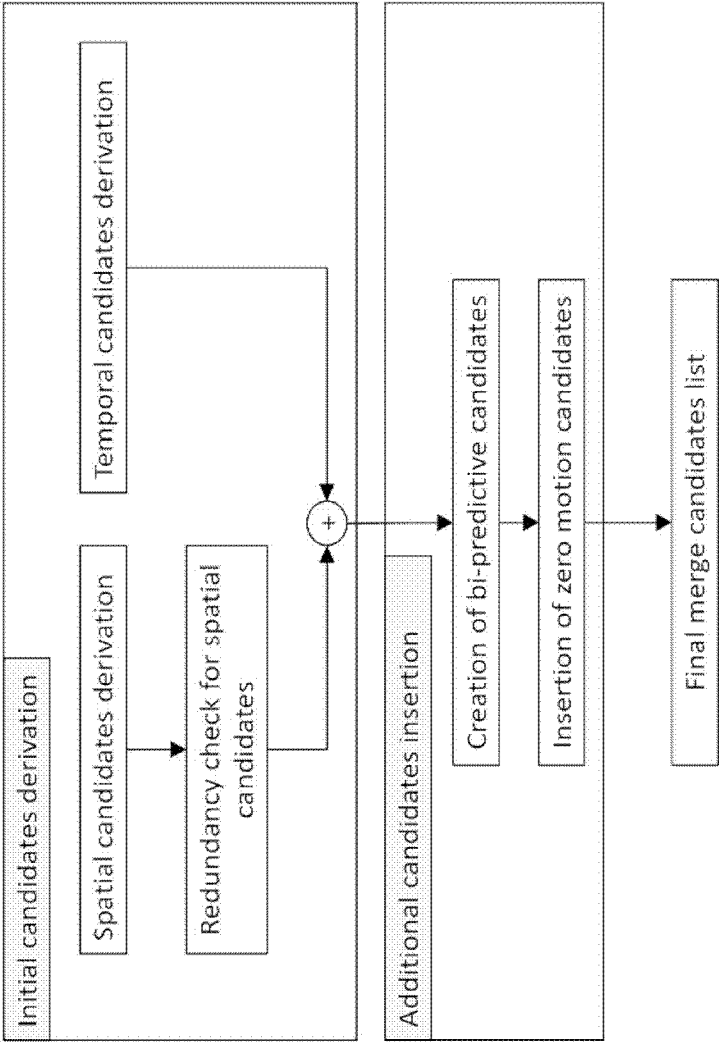


FIG. 1

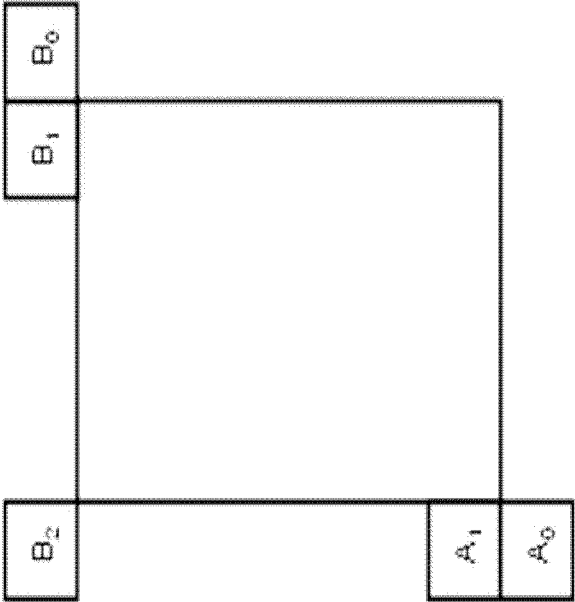


FIG. 2

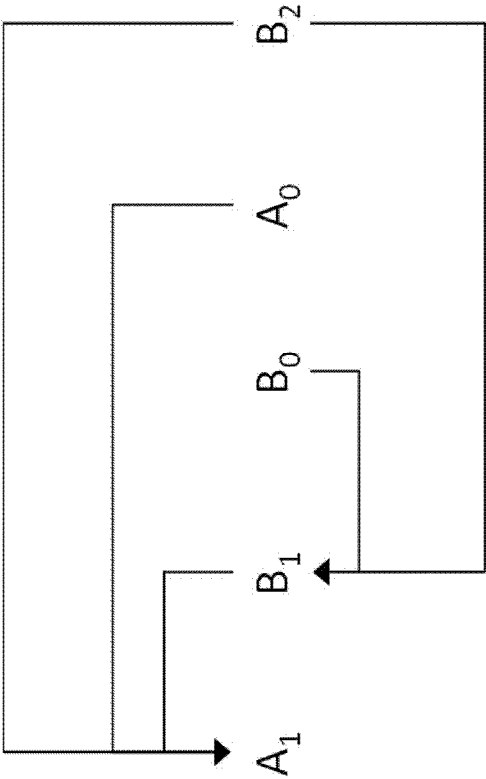


FIG. 3

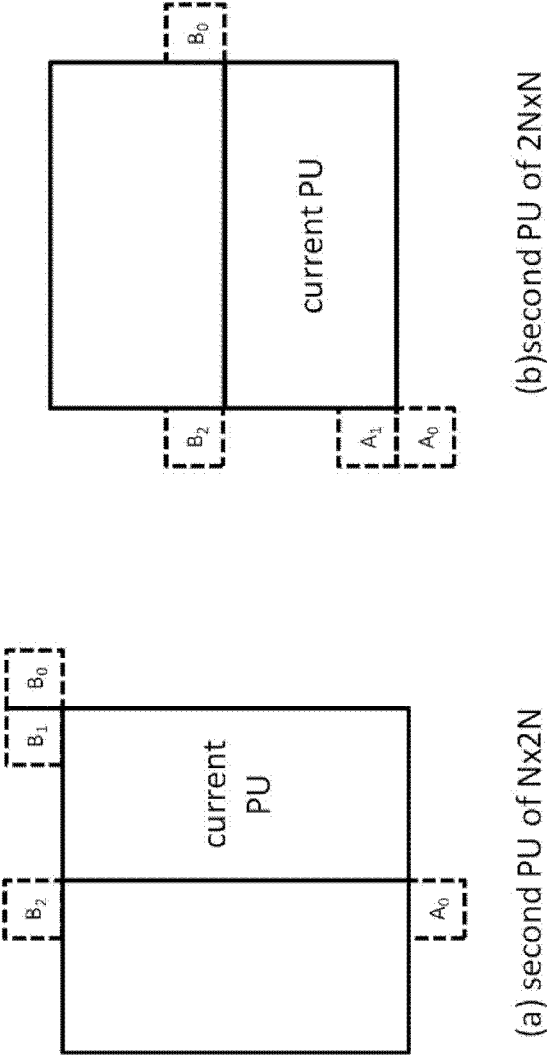


FIG. 4

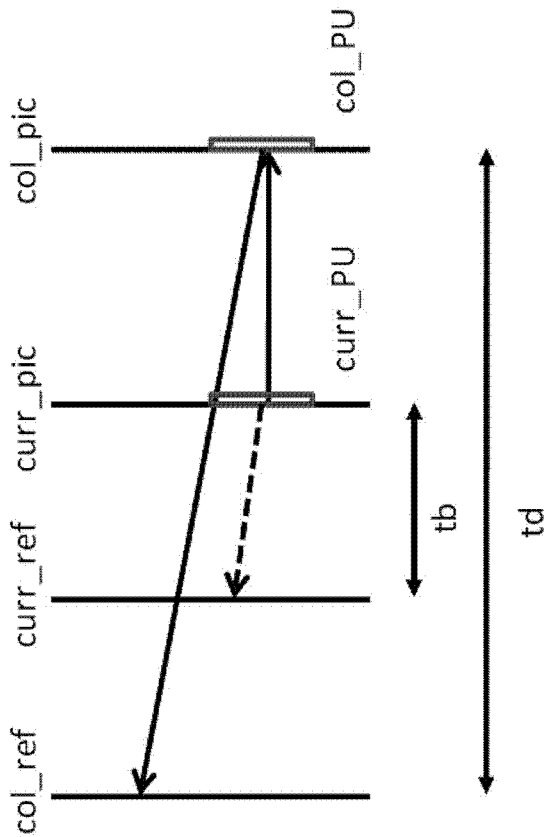


FIG. 5

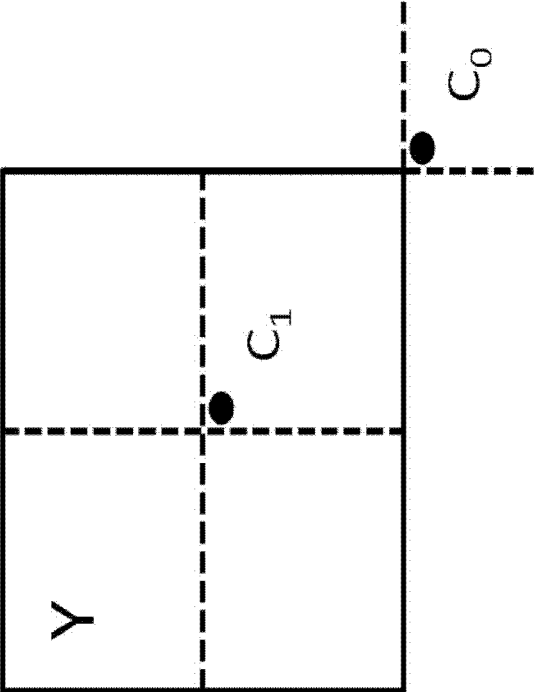


FIG. 6

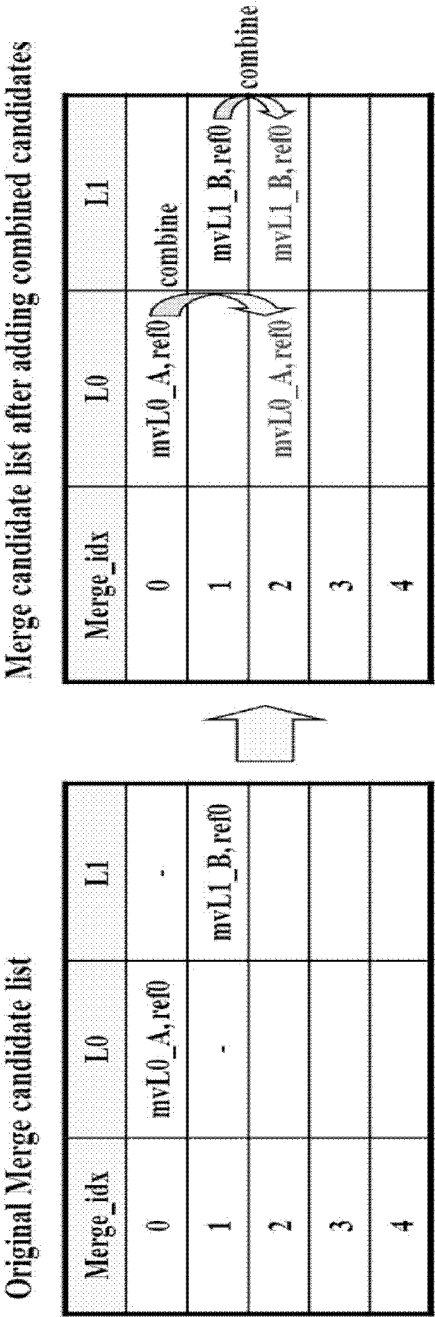


FIG. 7

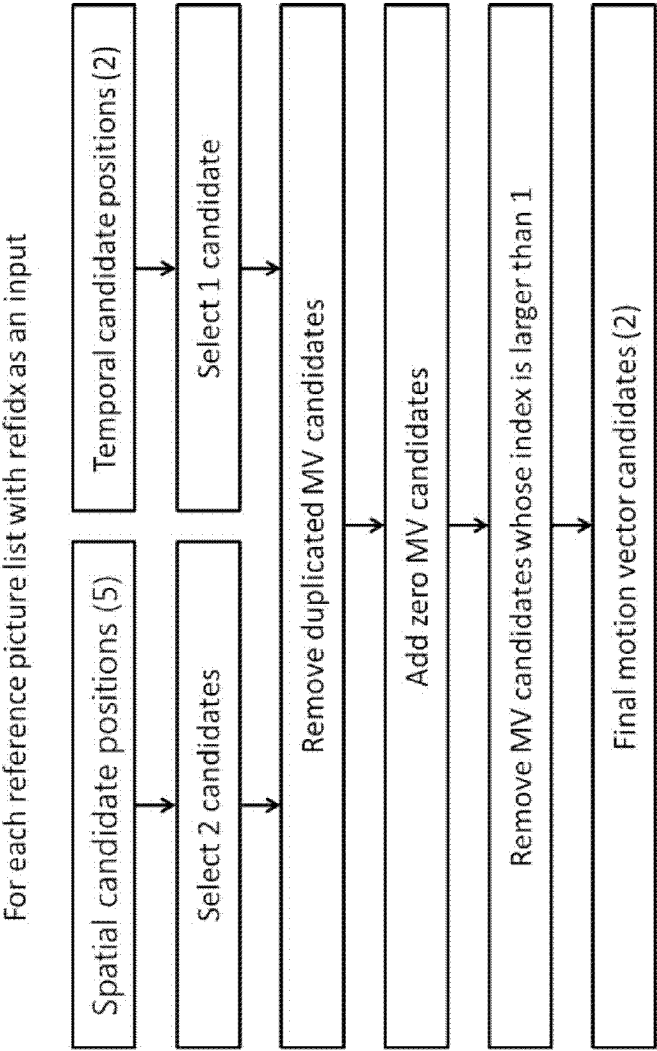


FIG. 8

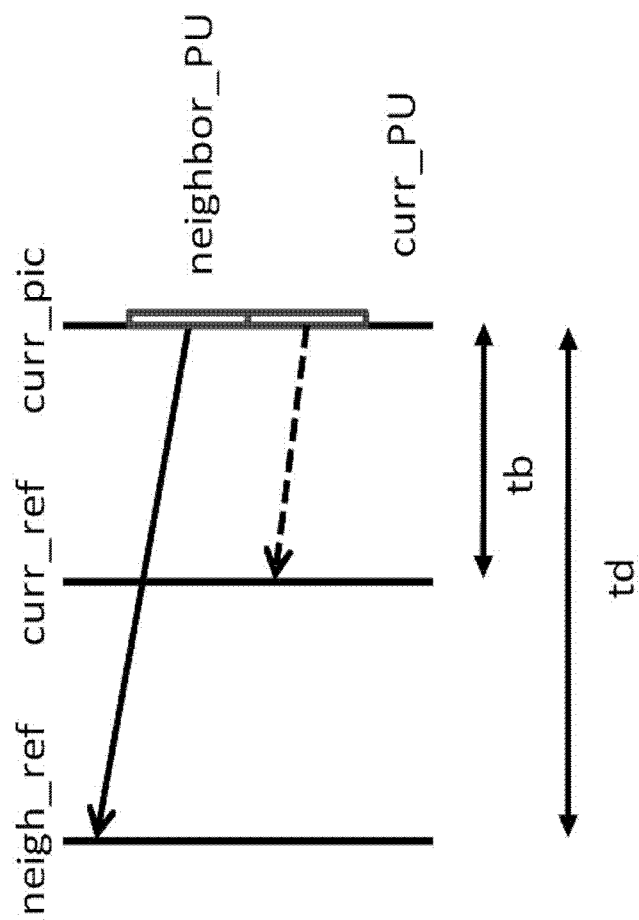


FIG. 9

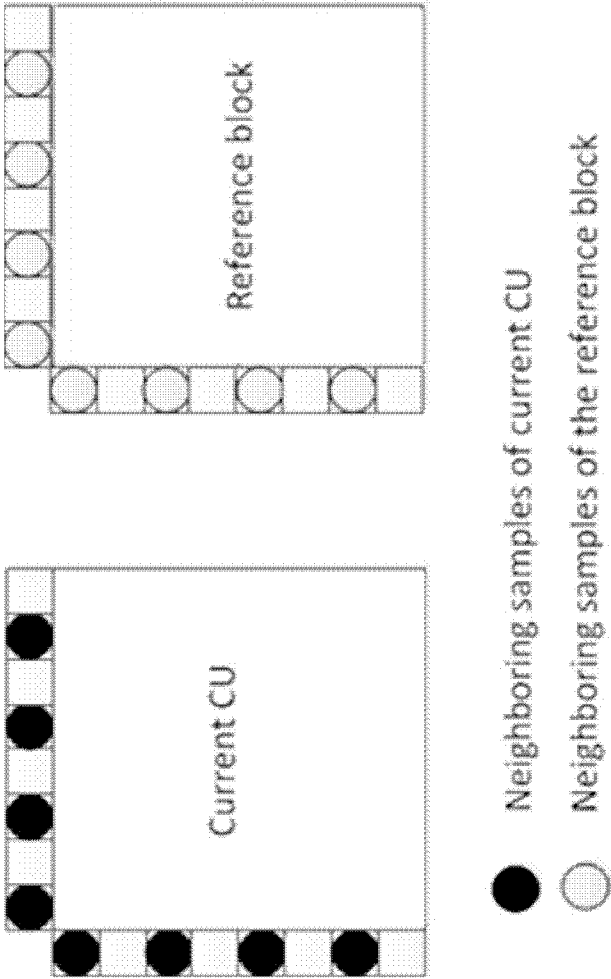


FIG. 10

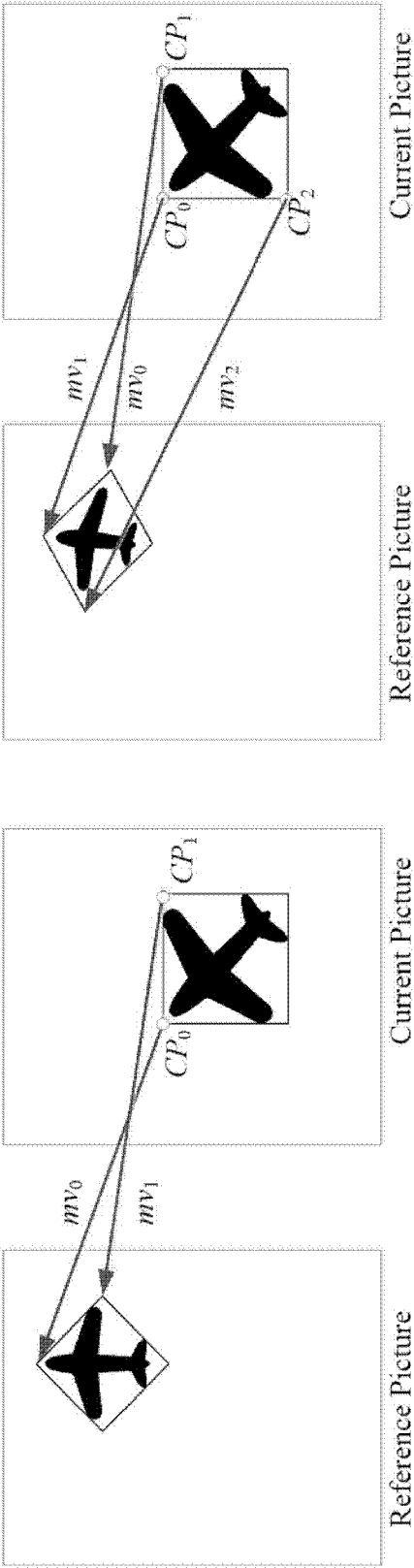


FIG. 11

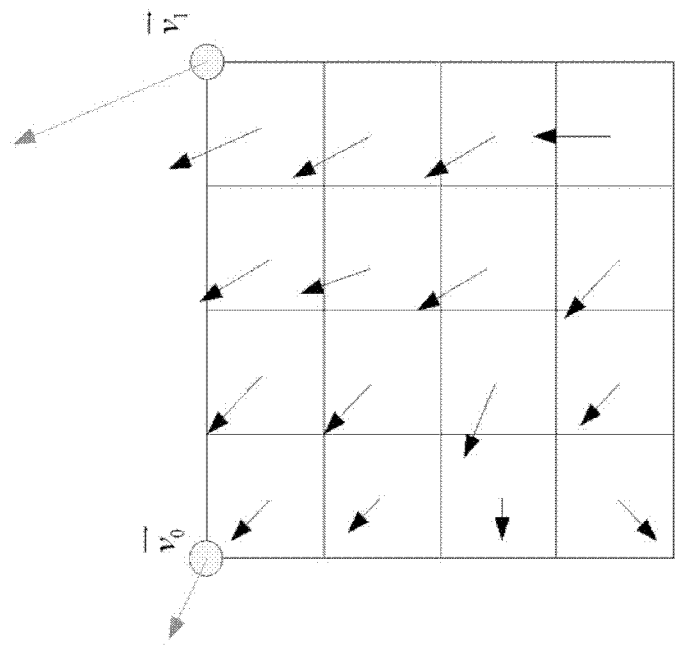


FIG. 12

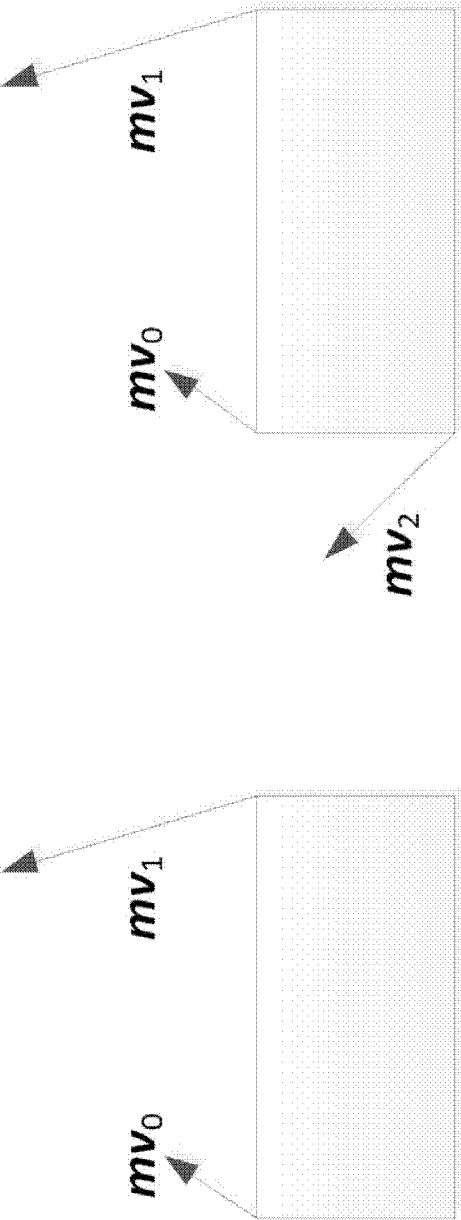


FIG. 13

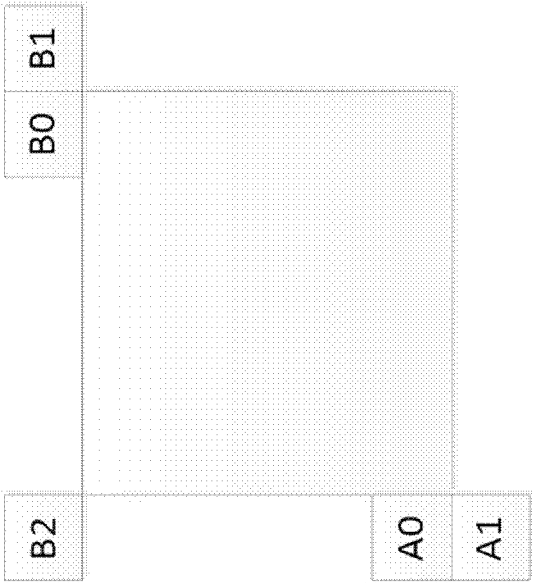


FIG. 14

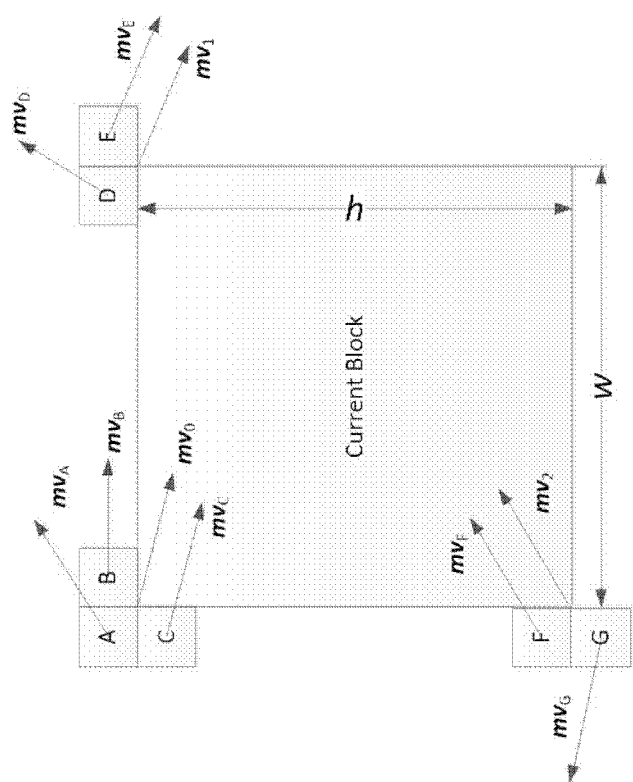


FIG. 15

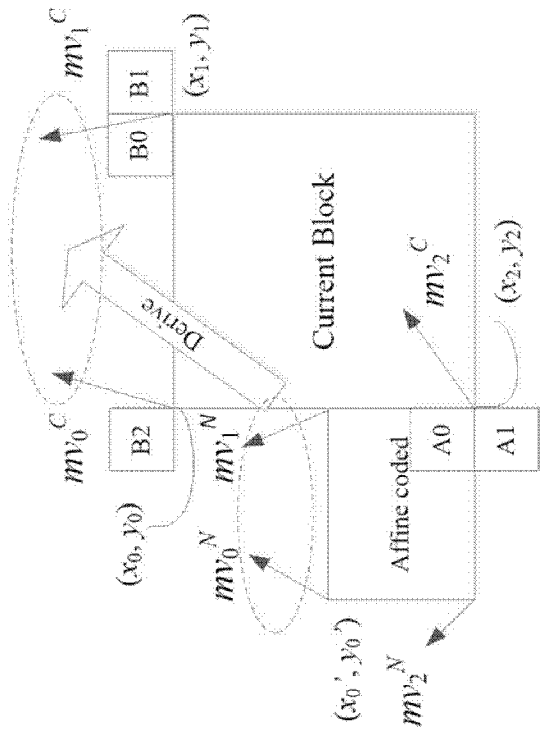
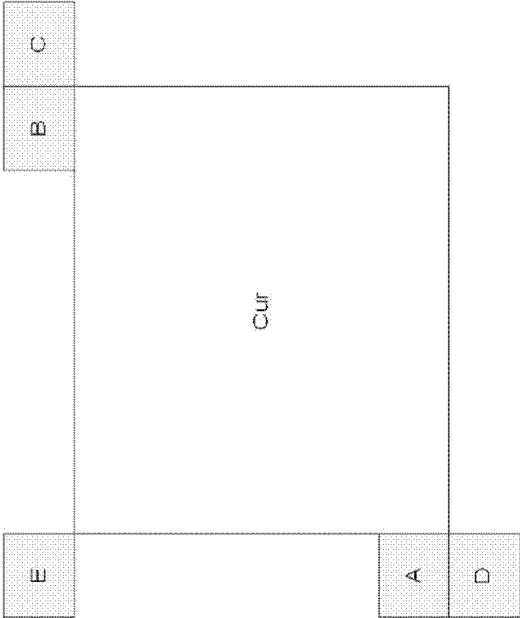


FIG. 16

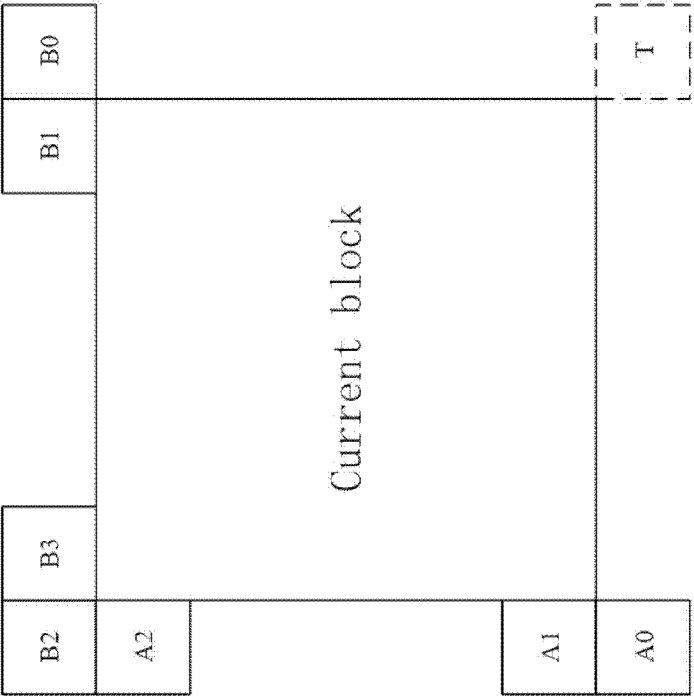


FIG. 17

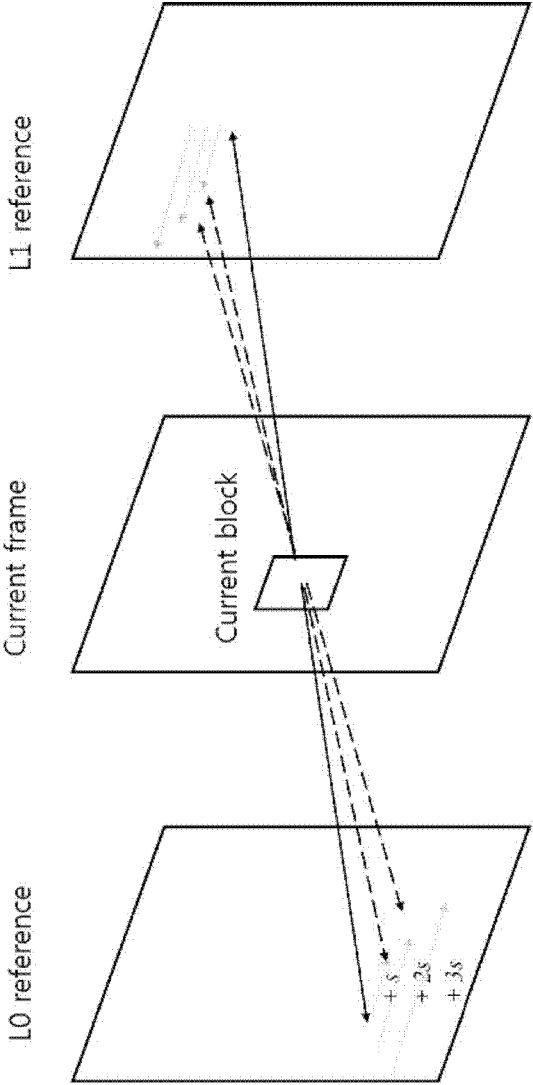


FIG. 18

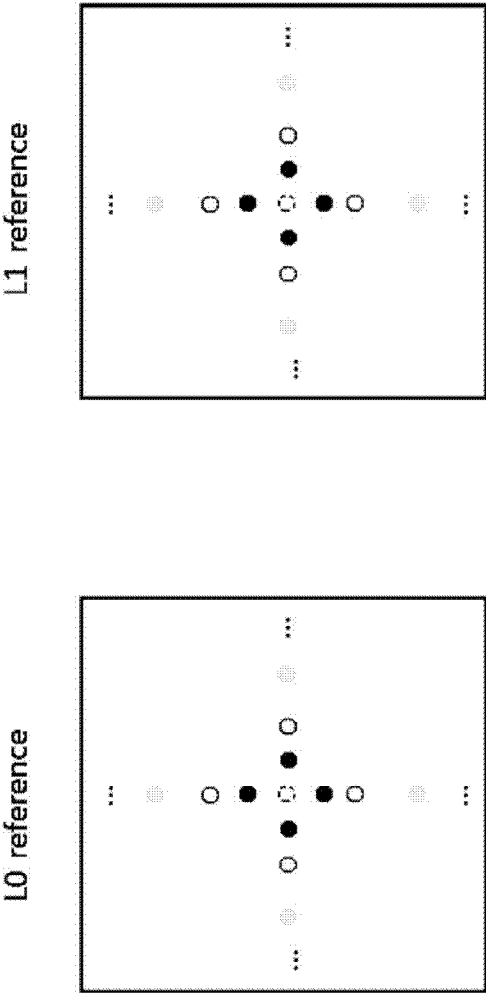


FIG. 19

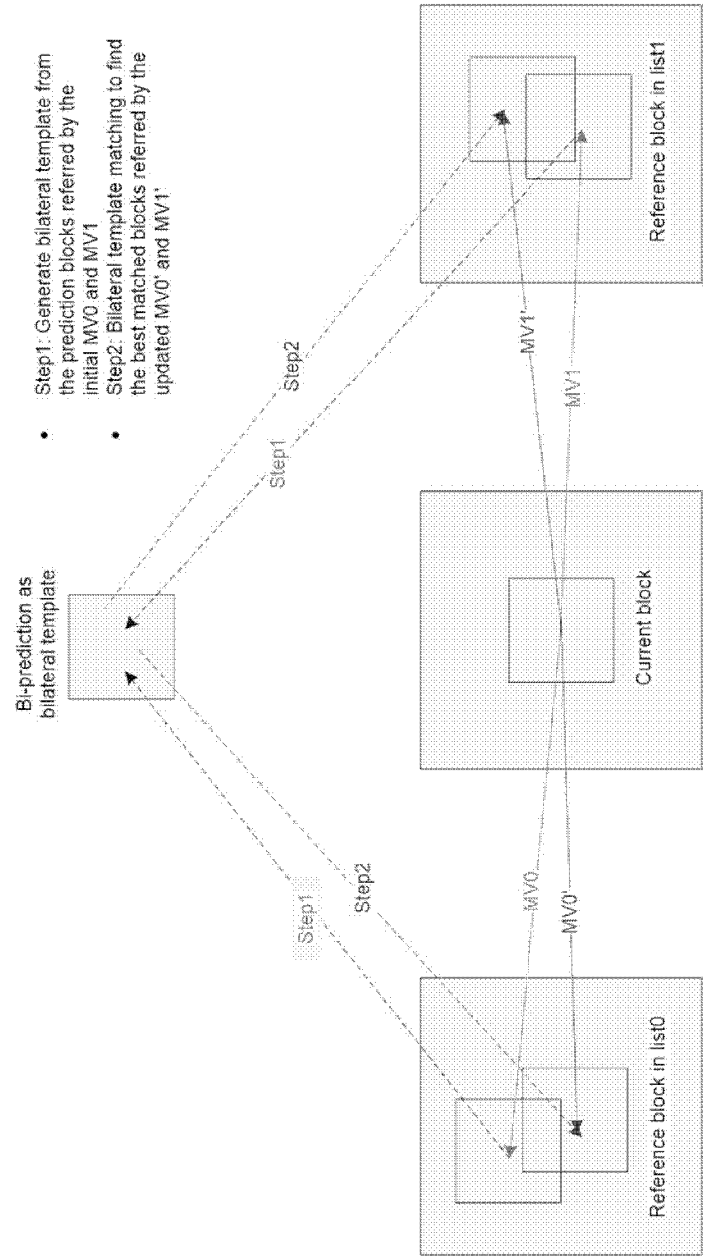


FIG. 20

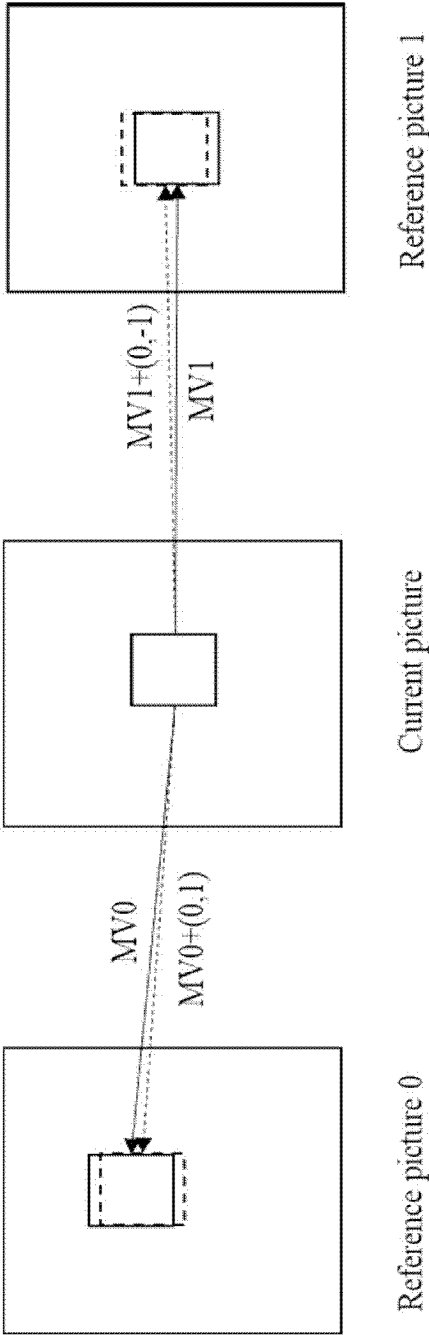


FIG. 21

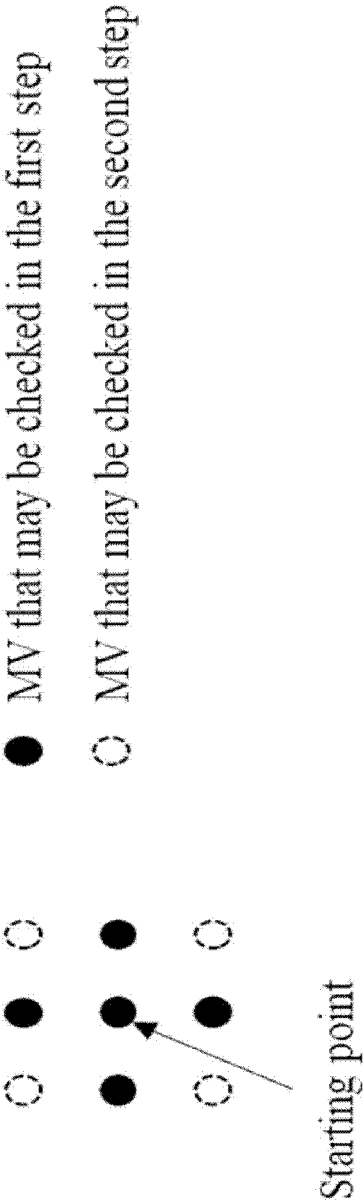


FIG. 22

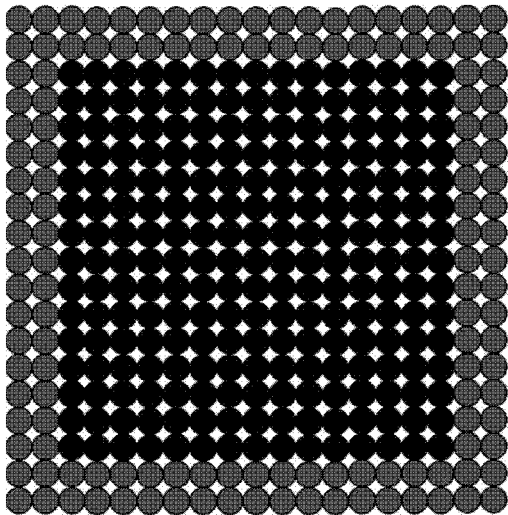


FIG. 23

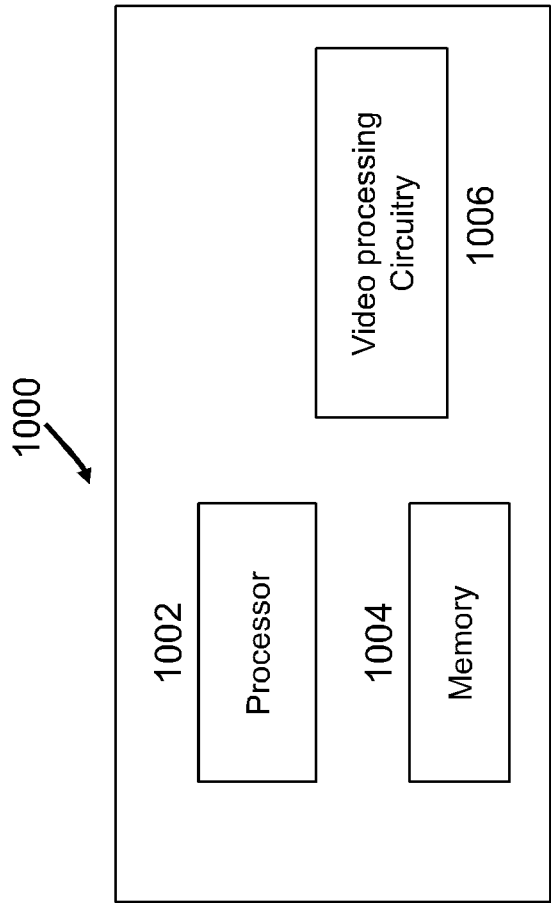


FIG. 24

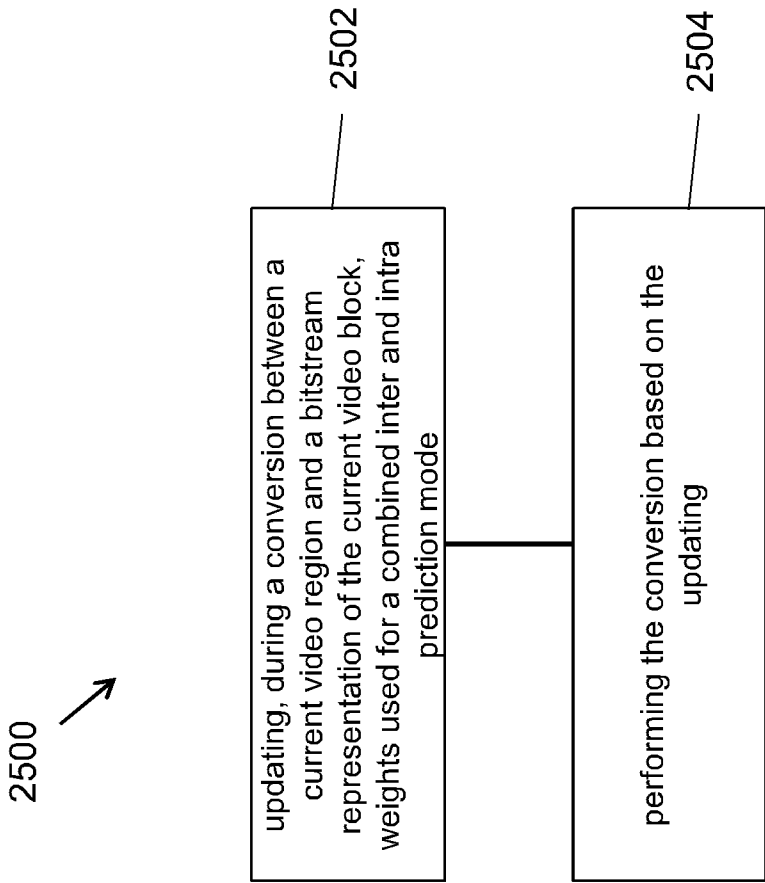


FIG. 25

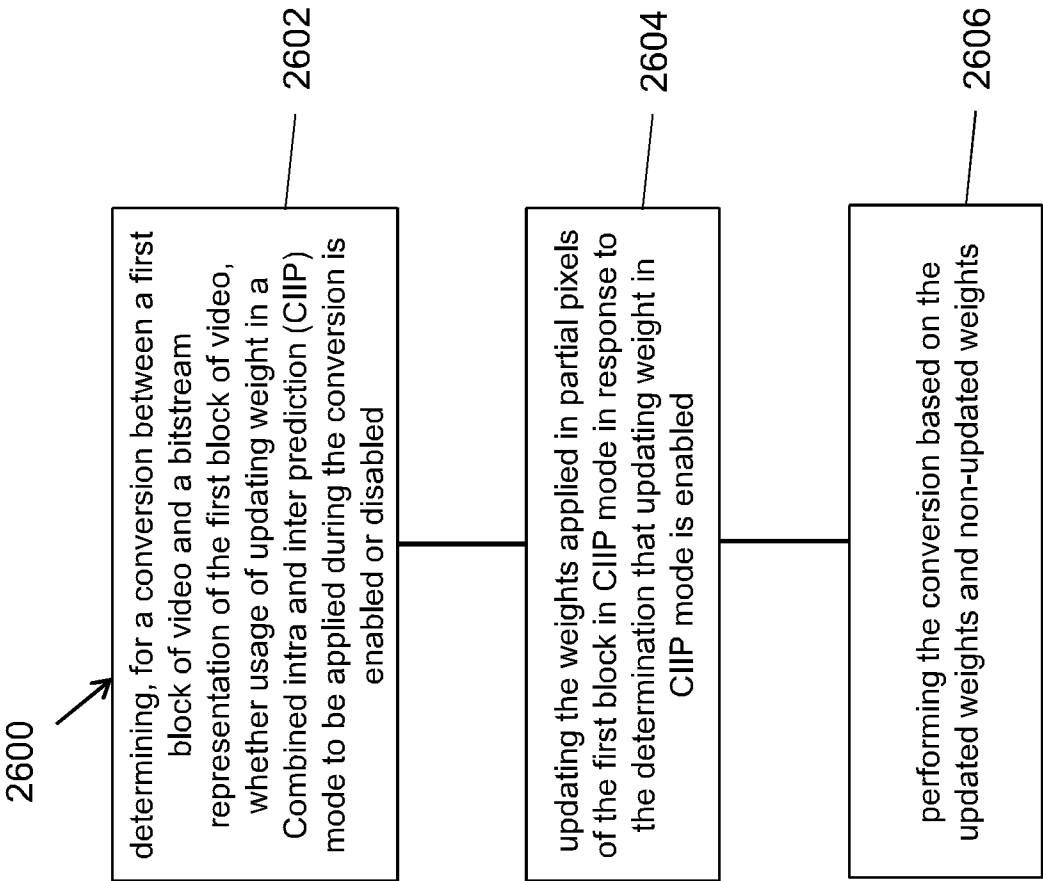


FIG. 26

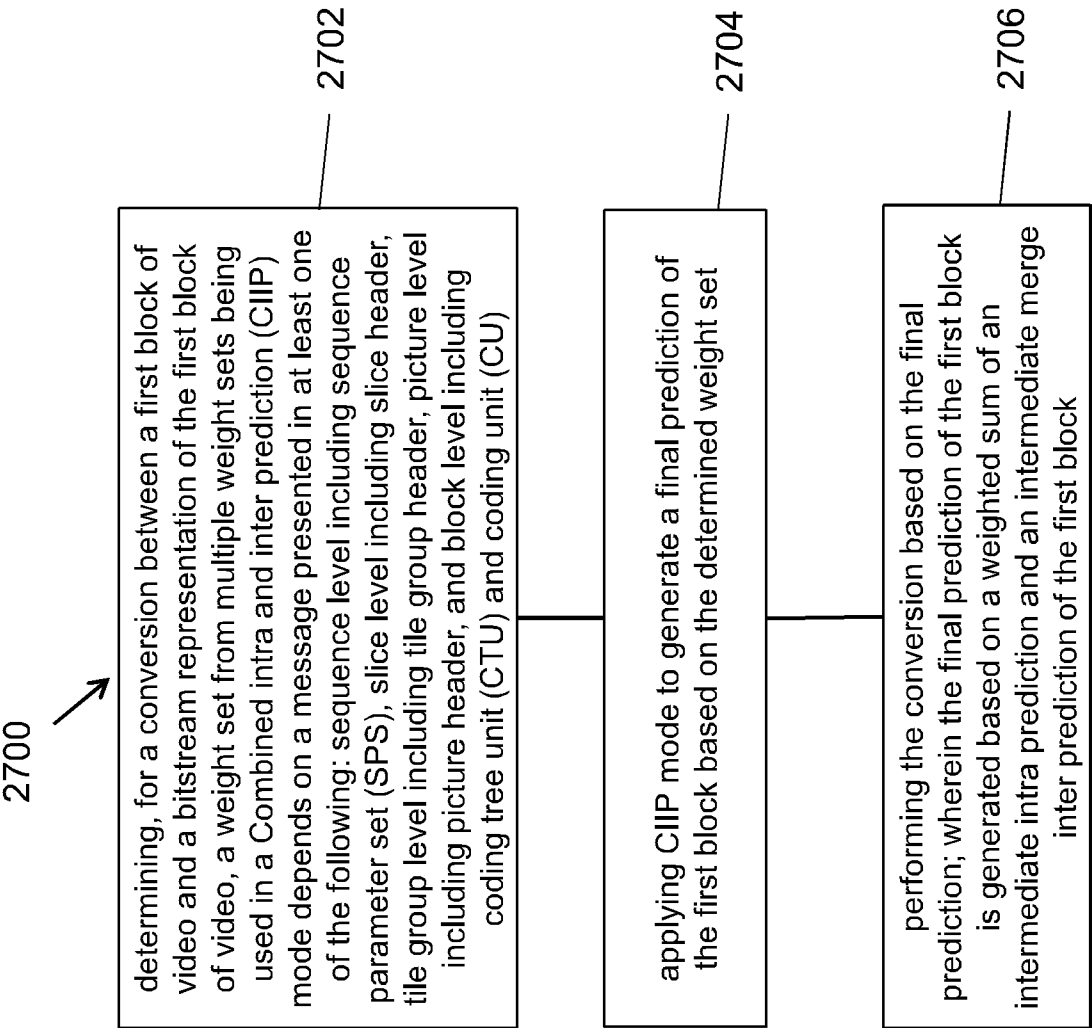


FIG. 27

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/078988

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/107(2014.01)i; H04N 19/176(2014.01)i; H04N 19/182(2014.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04N

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

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

CNPAT,CNKI,WPI,EPODOC,IEEE:video, bitstream, block, weight, set, sum, combine, intra, inter, predict, CIIP, pixel, merge, updat+, region

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2018288410 A1 (SAMSUNG ELECTRONICS CO., LTD.) 04 October 2018 (2018-10-04) claims 1-15, description, paragraphs [0060]-[0064], [0087]-[0117], [0122], [0135], [0142], [0143], and figures 3, 9	1, 3, 7-25, 36, 37
Y	US 2018288410 A1 (SAMSUNG ELECTRONICS CO., LTD.) 04 October 2018 (2018-10-04) claims 1-15, description, paragraphs [0060]-[0064], [0087]-[0117], [0122], [0135], [0142], [0143], and figures 3, 9	2, 4-6, 26-35
Y	ANDRE, Seixas Dias. "CE10-related: Multi-Hypothesis Intra with Weighted Combination of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 13th Meeting: Marrakech, MA, 9-18 Jan. 2019" <i>Joint Video Experts Team (JVET)</i> , No. JVET-M0454-v1, 18 January 2019 (2019-01-18), the abstract, parts 1 and 2	2
Y	CHIANG, Man-Shu. "CE10-related: Multi-Hypothesis Intra with Weighted Combination of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 13th Meeting: Marrakech, MA, 9-18 Jan. 2019" <i>Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 12th Meeting: Macao, CN, 3-12 Oct. 2018</i> , No. JVET-L0100-v3, 12 October 2018 (2018-10-12), part 2.3	4-6, 26-35

☒ Further documents are listed in the continuation of Box C.☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

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

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

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

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

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

12 May 2020

Date of mailing of the international search report

27 May 2020

Name and mailing address of the ISA/CN

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

Authorized officer

YU,Lina

Facsimile No. (86-10)62019451

Telephone No. (86-10)53961756

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/078988**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 3367681 A1 (LG ELECTRONICS INC.) 29 August 2018 (2018-08-29) the whole document	1-37
A	US 2018278942 A1 (QUALCOMM INCORPORATED) 27 September 2018 (2018-09-27) the whole document	1-37
A	US 2018249156 A1 (LG ELECTRONICS INC.) 30 August 2018 (2018-08-30) the whole document	1-37
A	CN 108702515 A (MEDIATEK INC.) 23 October 2018 (2018-10-23) the whole document	1-37

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2020/078988

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
US	2018288410	A1	04 October 2018	EP	3217663	A1	13 September 2017
				WO	2016072775	A1	12 May 2016
				CN	107113425	A	29 August 2017
				KR	20170084055	A	19 July 2017
				EP	3217663	A4	14 February 2018
EP	3367681	A1	29 August 2018	WO	2017069590	A1	27 April 2017
				US	2018309983	A1	25 October 2018
				KR	20180058224	A	31 May 2018
				CN	108293113	A	17 July 2018
				US	10595017	B2	17 March 2020
US	2018278942	A1	27 September 2018	EP	3367681	A4	22 May 2019
				SG	11201907091P	A	30 October 2019
				WO	2018175520	A1	27 September 2018
				CN	110393009	A	29 October 2019
				TW	201841508	A	16 November 2018
US	2018249156	A1	30 August 2018	AU	2018240212	A1	22 August 2019
				KR	20190127889	A	13 November 2019
				EP	3603061	A1	05 February 2020
				WO	2017043816	A1	16 March 2017
				KR	20180041211	A	23 April 2018
CN	108702515	A	23 October 2018	US	2017251213	A1	31 August 2017
				WO	2017144017	A1	31 August 2017
				TW	201735642	A	01 October 2017
				TW	I618402	B	11 March 2018
				GB	2564310	A	09 January 2019
				GB	201815108	D0	31 October 2018