



(51) International Patent Classification:

H04N 19/52 (2014.01)

(21) International Application Number:

PCT/CN2020/076981

(22) International Filing Date:

27 February 2020 (27.02.2020)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

PCT/CN2019/076297

27 February 2019 (27.02.2019) 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: **LIU, Hongbin**; Jinritoutiao Post Office, China Satellite Communications Tower, No.63, Zhichun Road,

Haidian District, Beijing 100080 (CN). **ZHANG, Li**; 12655 West Jefferson Boulevard, Sixth Floor, Suite No.137, Los Angeles, California 90066 (US). **ZHANG, Kai**; 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). **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).

(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,

(54) Title: MOTION VECTOR CODING FOR SCREEN CONTENT CODING

1700



Determining, based on a first indication in a bitstream representation of a current video block, whether a subpel motion precision for motion information is allowed for the current video block

1710

Performing, a conversion between the current video block and the bitstream representation of the current video block, using a non-subpel motion precision for the motion information after determining that the subpel motion precision is disallowed

1720

FIG. 17

(57) Abstract: The disclosure relates to motion vector coding for screen content coding. A method for video processing includes: determining, based on a first indication in a bitstream representation of a current video block, whether a subpel motion precision for motion information is allowed for the current video block, wherein motion information comprises at least one of a motion vector (MV), a motion vector prediction (MVP), and a motion vector difference (MVD); performing, a conversion between the current video block and the bitstream representation of the current video block, using a non-subpel motion precision for the motion information after determining that the subpel motion precision is disallowed.



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))*

## **MOTION VECTOR CODING FOR SCREEN CONTENT CODING**

### **CROSS REFERENCE TO RELATED APPLICATIONS**

**[001]** 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/CN2019/076297, filed on February 27, 2019. The entire disclosure of the aforementioned application is incorporated by reference as part of the disclosure of this application.

### **TECHNICAL FIELD**

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

### **BACKGROUND**

**[003]** 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**

**[004]** Devices, systems and methods related to digital video coding, and specifically, to motion vector coding for on-screen content coding vectors are described. The described methods may be applied to existing video coding standards (e.g., Versatile Video Coding) or the standard (Versatile Video Coding) to be finalized. It may be also applicable to future video coding standards or video codec.

**[005]** In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes determining that a member of the group including (i) a current tile, (ii) a current tile group, (iii) a current slice, or (iv) a current picture is associated with on-screen displayed content, and in response to the determining, enabling or disabling one or more sub-pixel motion vector parameters.

**[006]** In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes determining, based on a first indication in a bitstream representation of a current video block, whether a subpel motion precision for motion information is allowed for the current video block, wherein motion information comprises at least one of a motion vector (MV), a motion vector prediction (MVP), and a motion vector difference (MVD); performing, a conversion between the current video block and the bitstream representation of the current video block, using a non-subpel motion precision for the motion information after determining that the subpel motion precision is disallowed.

**[007]** In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes determining, during a conversion between a current video block and a bitstream representation of the current video block, a set of allowed motion precisions for motion information, wherein the current video block is coded in intra block copy (IBC) mode, wherein the motion information comprises at least one of a motion vector, a motion vector prediction and a motion vector difference; performing the conversion, based on the determined set of motion precisions.

**[008]** In another representative aspect, the above-described method is embodied in the form of processor-executable code and stored in a computer-readable program medium.

**[009]** In yet another representative aspect, an apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon is disclosed. The instructions upon execution by the processor, cause the processor to implement any one or more of the disclosed methods.

**[0010]** Further, in a representative aspect, any one or more of the disclosed methods is an encoder-side implementation.

**[0011]** Also, in a representative aspect, any one or more of the disclosed methods is a decoder-side implementation.

**[0012]** The above and other aspects and features of the disclosed technology are described in greater detail in the drawings, the description and the claims.

### BRIEF DESCRIPTION OF THE DRAWINGS

- [0013] FIG. 1 shows an example of an affine motion model.
- [0014] FIG. 2 shows an example of an affine motion vector field per sub-block.
- [0015] FIGs. 3A and 3B show a 4-parameter affine model and a 6-parameter affine model respectively.
- [0016] FIG. 4 shows an example motion vector prediction (MVP) for affine inter mode.
- [0017] FIG. 5A-5B shows example candidates for affine merge mode.
- [0018] FIG. 6 shows example candidate positions for an affine merge mode.
- [0019] FIG. 7 shows an example of a mapping between distance offset and distance mapping.
- [0020] FIG. 8 shows an example of an ultimate vector expression (UMVE) search process.
- [0021] FIG. 9 shows an example of a UMVE search point.
- [0022] FIG. 10 shows an example of intra block copy.
- [0023] FIG. 11 shows an example of positions of spatial candidates.
- [0024] FIG. 12 shows an example of candidate pairs subject to a redundancy check of spatial merge candidates.
- [0025] FIG. 13 shows an example of current coding tree unit (CTU) processing.
- [0026] FIG. 14 shows another example of CTU processing.
- [0027] FIG. 15 is a block diagram of an example of a hardware platform for implementing a visual media decoding or a visual media encoding technique described in the present document.
- [0028] FIG. 16 shows a flowchart of an example method for video coding.
- [0029] FIG. 17 shows a flowchart of an example method for video processing.
- [0030] FIG. 18 shows a flowchart of an example method for video processing.

## DETAILED DESCRIPTION

### 1. Video coding in HEVC

[0031] 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 Versatile Video Coding (VVC) standard targeting at 50% bitrate reduction compared to HEVC.

[0032] The latest version of VVC draft, i.e., Versatile Video Coding (Draft 4) could be found at [http://phenix.it-sudparis.eu/jvet/doc\\_end\\_user/documents/13\\_Marrakech/wg11/JVET-M1001-v3.zip](http://phenix.it-sudparis.eu/jvet/doc_end_user/documents/13_Marrakech/wg11/JVET-M1001-v3.zip). The latest reference software of VVC, named VTM, could be found at: [https://vcgit.hhi.fraunhofer.de/jvet/VVCSsoftware\\_VTM/tags/VTM-4.0](https://vcgit.hhi.fraunhofer.de/jvet/VVCSsoftware_VTM/tags/VTM-4.0)

### 2.1. Inter mode coding in HEVC

#### 7.3.6.1 General slice segment header syntax

| slice_segment_header() {                                                                          | Descriptor |
|---------------------------------------------------------------------------------------------------|------------|
| first_slice_segment_in_pic_flag                                                                   | ue(v)      |
| ...                                                                                               |            |
| if( slice_type == P    slice_type == B ) {                                                        |            |
| ...                                                                                               |            |
| if( ( weighted_pred_flag && slice_type == P )   <br>( weighted_bipred_flag && slice_type == B ) ) |            |
| pred_weight_table()                                                                               |            |
| <b>five_minus_max_num_merge_cand</b>                                                              | ue(v)      |
| <i>if( motion_vector_resolution_control_idc == 2 )</i>                                            |            |

| <i>use_integer_mv_flag</i> | <i>u(l)</i> |
|----------------------------|-------------|
| }                          |             |
| ...                        |             |
| }                          |             |

[0033] The above table represents the general slice segment header syntax. In this table, the italicized-bold text represents examples of new syntax elements for bitstream description. The syntax elements are also referred to herein as high-level syntax indicating whether sub-pixel precision is allowed (enabled) or not allowed (disabled).

[0034] *motion\_vector\_resolution\_control\_idc* controls the presence and inference of the *use\_integer\_mv\_flag* that specifies the resolution of motion vectors for inter prediction. The value of *motion\_vector\_resolution\_control\_idc* shall not be equal to 3 in bitstreams conforming to this version of this Specification. The value of 3 for *motion\_vector\_resolution\_control\_idc* is reserved for future use by ITU-T | ISO/IEC. When not present, the value of *motion\_vector\_resolution\_control\_idc* is inferred to be equal to 0.

[0035] *use\_integer\_mv\_flag* equal to 1 specifies that the resolution of motion vectors for inter prediction in the current slice is integer. *use\_integer\_mv\_flag* equal to 0 specifies that the resolution of motion vectors for inter prediction in the current slice that refer to pictures other than the current picture is fractional with quarter-sample precision in units of luma samples. When not present, the value of *use\_integer\_mv\_flag* is inferred to be equal to *motion\_vector\_resolution\_control\_idc*.

## 2.2. Affine motion compensation prediction

[0036] 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 other irregular motions. In the JEM, a simplified affine transform motion compensation prediction is applied. As shown in FIG. 1, the affine motion field of the block is described by two control point motion vectors.

[0037] The motion vector field (MVF) of a block is described by the following equation:

$$\begin{cases} v_x = \frac{(v_{1x} - v_{0x})}{w}x - \frac{(v_{1y} - v_{0y})}{w}y + v_{0x} \\ v_y = \frac{(v_{1y} - v_{0y})}{w}x + \frac{(v_{1x} - v_{0x})}{w}y + v_{0y} \end{cases} \quad (1)$$

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

[0038] In order to further simplify the motion compensation prediction, sub-block based affine transform prediction is applied. The sub-block size  $M \times N$  is derived as in Equation 2, where  $MvPre$  is the motion vector fraction accuracy (1/16 in JEM),  $(v_{2x}, v_{2y})$  is motion vector of the bottom-left control point, calculated according to Equation 1.

$$\begin{cases} M = clip3(4, w, \frac{w \times MvPre}{\max(abs(v_{1x} - v_{0x}), abs(v_{1y} - v_{0y}))}) \\ N = clip3(4, h, \frac{h \times MvPre}{\max(abs(v_{2x} - v_{0x}), abs(v_{2y} - v_{0y}))}) \end{cases} \quad (2)$$

[0039] After derived by Equation 2, M and N should be adjusted downward if necessary to make it a divisor of w and h, respectively.

[0040] To derive motion vector of each  $M \times N$  sub-block, the motion vector of the center sample of each sub-block, as shown in FIG. 2, is calculated according to Equation 1, and rounded to 1/16 fraction accuracy.

[0041] After MCP, the high accuracy motion vector of each sub-block is rounded and saved as the same accuracy as the normal motion vector.

### 2.2.1. AF\_INTER mode

[0042] In the JEM, there are two affine motion modes: AF\_INTER mode and AF\_MERGE mode. 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. In this mode, a candidate list with motion vector pair  $\{(v_0, v_1) | v_0 = \{v_A, v_B, v_C\}, v_1 = \{v_D, v_E\}\}$  is constructed using the neighbour blocks. As shown in FIG. 4,  $v_0$  is selected from the motion vectors of the block A, B or C. The motion vector from the neighbour block is scaled according to the reference list and the relationship among the picture order count (POC) of the reference for the neighbour block, the POC of the reference for the current CU and the POC of the current CU. And the approach to select  $v_1$  from the neighbour block D and E is similar. If the number of candidate list is smaller than 2, the list is padded by the motion vector pair composed by duplicating each of the AMVP candidates. When the candidate list is larger than 2, the candidates are firstly sorted according

to the consistency of the neighbouring motion vectors (similarity of the two motion vectors in a pair candidate) and only the first two candidates are kept. An RD cost check is used to determine which motion vector pair candidate is selected as the control point motion vector prediction (CPMVP) of the current CU. And an index indicating the position of the CPMVP in the candidate list is signalled in the bitstream. After the CPMVP of the current affine CU is determined, affine motion estimation is applied and the control point motion vector (CPMV) is found. Then the difference of the CPMV and the CPMVP is signalled in the bitstream.

**[0043]** In AF\_INTER mode, when 4/6 parameter affine mode is used, 2/3 control points can be used, and therefore 2/3 MVD needs to be coded for these control points, as shown in FIG. 3A. In JVET-K0337, it is proposed to derive the MV as follows, i.e.,  $mvd_1$  and  $mvd_2$  are predicted from  $mvd_0$ .

$$\begin{aligned} mv_0 &= \overline{mv}_0 + mvd_0 \\ mv_1 &= \overline{mv}_1 + mvd_1 + mvd_0 \\ mv_2 &= \overline{mv}_2 + mvd_2 + mvd_0 \end{aligned}$$

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. 3B. 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.

### 2.2.2. Fast affine ME algorithm in AF\_INTER mode

**[0044]** In affine mode, MV of 2 or 3 control points needs to be determined jointly. Directly searching the multiple MVs jointly is computationally complex. A fast affine ME algorithm is proposed and is adopted into VTM/BMS.

**[0045]** The fast affine ME algorithm is described for the 4-parameter affine model, and the idea can be extended to 6-parameter affine model.

$$\begin{cases} x' = ax + by + c \\ y' = -bx + ay + d \end{cases} \quad (3)$$

$$\begin{cases} mv_{(x,y)}^h = x' - x = (a - 1)x + by + c \\ mv_{(x,y)}^v = y' - y = -bx + (a - 1)y + d \end{cases} \quad (4)$$

[0046] Replace  $(a - 1)$  with  $a'$ , then the motion vector can be rewritten as:

$$\begin{cases} mv_{(x,y)}^h = x' - x = a'x + by + c \\ mv_{(x,y)}^v = y' - y = -bx + a'y + d \end{cases} \quad (5)$$

[0047] Suppose motion vectors of the two controls points  $(0, 0)$  and  $(0, w)$  are known, from Equation (5), the affine parameters can be derived as follows:

$$\begin{cases} c = mv_{(0,0)}^h \\ d = mv_{(0,0)}^v \end{cases} \quad (6)$$

[0048] The motion vectors can be rewritten in vector form as:

$$MV(p) = A(P) * MV_C^T \quad (7)$$

Wherein

$$A(P) = \begin{bmatrix} 1 & x & 0 & y \\ 0 & y & 1 & -x \end{bmatrix} \quad (8)$$

$$MV_C = [mv_{(0,0)}^h \ a \ mv_{(0,0)}^v \ b] \quad (9)$$

$P = (x, y)$  is the pixel position.

[0049] At encoder, MVD of AF\_INTER are derived iteratively. Denote  $MV^i(P)$  as the MV derived in the  $i$ th iteration for position  $P$  and denote  $dMV_C^i$  as the delta updated for  $MV_C$  in the  $i$ th iteration. Then in the  $(i+1)$ th iteration,

$$\begin{aligned} MV^{i+1}(P) &= A(P) * \left( (MV_C^i)^T + (dMV_C^i)^T \right) \\ &= A(P) * (MV_C^i)^T + A(P) * (dMV_C^i)^T \\ &= MV^i(P) + A(P) * (dMV_C^i)^T \end{aligned} \quad (10)$$

[0050] Denote  $Pic_{ref}$  as the reference picture and denote  $Pic_{cur}$  as the current picture and denote  $Q = P + MV^i(P)$ . Suppose MSE is used as the matching criterion, then the following expression can be written:

$$\min \sum_P \left( Pic_{cur}(P) - Pic_{ref}(P + MV^{i+1}(P)) \right)^2$$

$$= \min \sum_P \left( Pic_{cur}(P) - Pic_{ref} \left( Q + A(P) * (dMV_C^i)^T \right) \right)^2 \quad (11)$$

Suppose  $(dMV_C^i)^T$  is small enough, then  $Pic_{ref} \left( Q + A(P) * (dMV_C^i)^T \right)$  can be written approximately as follows with 1th order Taylor expansion.

$$Pic_{ref} \left( Q + A(P) * (dMV_C^i)^T \right) \approx Pic_{ref}(Q) + Pic_{ref}'(Q) * A(P) * (dMV_C^i)^T \quad (12)$$

Wherein  $Pic_{ref}'(Q) = \left[ \frac{dPic_{ref}(Q)}{dx} \quad \frac{dPic_{ref}(Q)}{dy} \right]$ . Denote  $E^{i+1}(P) = Pic_{cur}(P) - Pic_{ref}(Q)$ ,

$$\begin{aligned} \min \sum_P \left( Pic_{cur}(P) - Pic_{ref}(Q) - Pic_{ref}'(Q) * A(P) * (dMV_C^i)^T \right)^2 \\ \min \sum_P \left( Pic_{cur}(P) - Pic_{ref}(Q) - Pic_{ref}'(Q) * A(P) * (dMV_C^i)^T \right)^2 \end{aligned} \quad (13)$$

$dMV_C^i$  can be derived by setting the derivative of the error function to zero. Then can then calculate delta MV of the control points (0, 0) and (0, w) according to  $A(P) * (dMV_C^i)^T$ ,

$$dMV_{(0,0)}^h = dMV_C^i[0] \quad (14)$$

$$dMV_{(0,w)}^h = dMV_C^i[1] * w + dMV_C^i[2] \quad (15)$$

$$dMV_{(0,0)}^v = dMV_C^i[2] \quad (16)$$

$$dMV_{(0,w)}^v = -dMV_C^i[3] * w + dMV_C^i[2] \quad (17)$$

**[0051]** Suppose such MVD derivation process is iterated by n times, then the final MVD is calculated as follows,

$$dMV_{(0,w)}^v = -dMV_C^i[3] * w + dMV_C^i[2] \quad (18)$$

$$dMV_{(0,w)}^v = -dMV_C^i[3] * w + dMV_C^i[2] \quad (19)$$

$$dMV_{(0,w)}^v = -dMV_C^i[3] * w + dMV_C^i[2] \quad (20)$$

$$fdMV_{(0,w)}^v = \sum_{i=0}^{n-1} -dMV_C^i[3] * w + \sum_{i=0}^{n-1} dMV_C^i[2] \quad (21)$$

[0052] In some embodiments, i.e., predicting delta MV of control point (0, w), denoted by  $mvd_1$  from delta MV of control point (0, 0), denoted by  $mvd_0$ , now actually only  $(\sum_{i=0}^{n-1} dMV_C^i[1] * w, -\sum_{i=0}^{n-1} dMV_C^i[3] * w)$  is encoded form  $mvd_1$ .

### 2.2.3. AF\_MERGE mode

[0053] When a CU is applied in affine merge (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. 5A. If the neighbour left bottom block A is coded in affine mode as shown in FIG. 5B, the motion vectors  $v_2$ ,  $v_3$  and  $v_4$  of the top left corner, above right corner and left bottom corner of the CU which contains the block A are derived. The motion vector  $v_0$  of the top left corner on the current CU is calculated according to  $v_2$ ,  $v_3$  and  $v_4$ . Secondly, the motion vector  $v_1$  of the above right of the current CU is calculated.

[0054] After the CPMV of the current CU  $v_0$  and  $v_1$  are derived, according to the simplified affine motion model Equation 1, 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.

[0055] In some embodiments, an affine merge candidate list is constructed with following steps:

[0056] 1) Insert inherited affine candidates

[0057] Inherited affine candidate means that the candidate is derived from the affine motion model of its valid neighbor affine coded block. In the common base, as shown in FIG. 6, the scan order for the candidate positions is: A1, B1, B0, A0 and B2.

[0058] 2) Insert constructed affine candidates

[0059] If the number of candidates in affine merge candidate list is less than MaxNumAffineCand (set to 5 in this contribution), 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.

**[0060]** The motion information for the control points is derived firstly from the specified spatial neighbors and temporal neighbor shown in FIG. 6. CP<sub>k</sub> (k=1, 2, 3, 4) represents the k-th control point. A<sub>0</sub>, A<sub>1</sub>, A<sub>2</sub>, B<sub>0</sub>, B<sub>1</sub>, B<sub>2</sub> and B<sub>3</sub> are spatial positions for predicting CP<sub>k</sub> (k=1, 2, 3); T is temporal position for predicting CP<sub>4</sub>.

**[0061]** The coordinates of CP<sub>1</sub>, CP<sub>2</sub>, CP<sub>3</sub> and CP<sub>4</sub> is (0, 0), (W, 0), (H, 0) and (W, H), respectively, where W and H are the width and height of current block.

**[0062]** The motion information of each control point is obtained according to the following priority order:

**[0063]** For CP<sub>1</sub>, the checking priority is B<sub>2</sub>->B<sub>3</sub>->A<sub>2</sub>. B<sub>2</sub> is used if it is available. Otherwise, if B<sub>2</sub> is available, B<sub>3</sub> is used. If both B<sub>2</sub> and B<sub>3</sub> are unavailable, A<sub>2</sub> is used. If all the three candidates are unavailable, the motion information of CP<sub>1</sub> cannot be obtained.

**[0064]** For CP<sub>2</sub>, the checking priority is B<sub>1</sub>->B<sub>0</sub>.

**[0065]** For CP<sub>3</sub>, the checking priority is A<sub>1</sub>->A<sub>0</sub>.

**[0066]** For CP<sub>4</sub>, T is used.

**[0067]** Secondly, the combinations of controls points are used to construct an affine merge candidate.

**[0068]** 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 ({CP<sub>1</sub>, CP<sub>2</sub>, CP<sub>4</sub>}, {CP<sub>1</sub>, CP<sub>2</sub>, CP<sub>3</sub>}, {CP<sub>2</sub>, CP<sub>3</sub>, CP<sub>4</sub>}, {CP<sub>1</sub>, CP<sub>3</sub>, CP<sub>4</sub>}). Combinations {CP<sub>1</sub>, CP<sub>2</sub>, CP<sub>3</sub>}, {CP<sub>2</sub>, CP<sub>3</sub>, CP<sub>4</sub>}, {CP<sub>1</sub>, CP<sub>3</sub>, CP<sub>4</sub>} will be converted to a 6-parameter motion model represented by top-left, top-right and bottom-left control points.

**[0069]** 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 following six combinations ({CP<sub>1</sub>, CP<sub>4</sub>}, {CP<sub>2</sub>, CP<sub>3</sub>}, {CP<sub>1</sub>, CP<sub>2</sub>}, {CP<sub>2</sub>, CP<sub>4</sub>}, {CP<sub>1</sub>, CP<sub>3</sub>}, {CP<sub>3</sub>, CP<sub>4</sub>}). Combinations {CP<sub>1</sub>, CP<sub>4</sub>}, {CP<sub>2</sub>, CP<sub>3</sub>}, {CP<sub>2</sub>, CP<sub>4</sub>}, {CP<sub>1</sub>, CP<sub>3</sub>}, {CP<sub>3</sub>, CP<sub>4</sub>} will be converted to a 4-parameter motion model represented by top-left and top-right control points.

**[0070]** 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}, {CP2, CP3}, {CP1, CP4}, {CP2, CP4}, {CP3, CP4}

[0071] For reference list X (X being 0 or 1) of a combination, the reference index with highest usage ratio in the control points is selected as the reference index of list X, and motion vectors point to difference reference picture will be scaled.

[0072] After a candidate is derived, full pruning process is performed to check whether same candidate has been inserted into the list. If a same candidate exists, the derived candidate is discarded.

[0073] 3) Padding with zero motion vectors

[0074] If the number of candidates in affine merge candidate list is less than 5, zero motion vectors with zero reference indices are inserted into the candidate list, until the list is full.

### 2.3. Affine merge mode with prediction offsets

[0075] In some embodiments, UMVE is extended to affine merge mode. In this document, the term “UMVE” and “affine mode” are used interchangeably. The proposed method selects the first available affine merge candidate as a base predictor. Then it applies a motion vector offset to each control point’s motion vector value from the base predictor. If there is no affine merge candidate available, this proposed method will not be used.

[0076] The selected base predictor’s inter prediction direction, and the reference index of each direction is used without change.

[0077] In the current implementation, the current block’s affine model is assumed to be a 4-parameter model, only 2 control points need to be derived. Thus, only the first 2 control points of the base predictor will be used as control point predictors.

[0078] For each control point, a zero\_MVD flag is used to indicate whether the control point of current block has the same MV value as the corresponding control point predictor. If zero\_MVD flag is true, there’s no other signaling needed for the control point. Otherwise, a distance index and an offset direction index is signaled for the control point.

[0079] A distance offset table with size of 5 is used as shown in the table below. Distance index is signaled to indicate which distance offset to use. The mapping of distance index and distance offset values is shown in FIG. 7.

Table 1. Distance offset table

| Distance IDX    | 0       | 1     | 2     | 3     | 4     |
|-----------------|---------|-------|-------|-------|-------|
| Distance-offset | 1/2-pel | 1-pel | 2-pel | 4-pel | 8-pel |

[0080] The direction index can represent four directions as shown below, where only x or y direction may have an MV difference, but not in both directions.

| Offset Direction IDX | 00 | 01 | 10 | 11 |
|----------------------|----|----|----|----|
| x-dir-factor         | +1 | -1 | 0  | 0  |
| y-dir-factor         | 0  | 0  | +1 | -1 |

[0081] If the inter prediction is uni-directional, the signaled distance offset is applied on the offset direction for each control point predictor. Results will be the MV value of each control point.

[0082] For example, when base predictor is uni-directional, and the motion vector values of a control point is MVP ( $v_{px}$ ,  $v_{py}$ ). When distance offset and direction index are signaled, the motion vectors of current block's corresponding control points will be calculated as below.

$$MV(v_x, v_y) = MVP(v_{px}, v_{py}) + MV(x\text{-dir-factor} * \text{distance-offset}, y\text{-dir-factor} * \text{distance-offset});$$

[0083] If the inter prediction is bi-directional, the signaled distance offset is applied on the signaled offset direction for control point predictor's L0 motion vector; and the same distance offset with opposite direction is applied for control point predictor's L1 motion vector. Results will be the MV values of each control point, on each inter prediction direction.

[0084] For example, when base predictor is uni-directional, and the motion vector values of a control point on L0 is MVP<sub>L0</sub> ( $v_{0px}$ ,  $v_{0py}$ ), and the motion vector of that control point on L1 is MVP<sub>L1</sub> ( $v_{1px}$ ,  $v_{1py}$ ). When distance offset and direction index are signaled, the motion vectors of current block's corresponding control points will be calculated as below.

$MV_{L0}(v_{0x}, v_{0y}) = MVP_{L0}(v_{0px}, v_{0py}) + MV(x\text{-dir-factor} * \text{distance-offset}, y\text{-dir-factor} * \text{distance-offset});$

$MV_{L1}(v_{0x}, v_{0y}) = MVP_{L1}(v_{0px}, v_{0py}) + MV(-x\text{-dir-factor} * \text{distance-offset}, -y\text{-dir-factor} * \text{distance-offset});$

## 2.4. Ultimate motion vector expression

[0085] In some embodiments, ultimate motion vector expression (UMVE) is presented. UMVE is used for either skip or merge modes with a proposed motion vector expression method.

[0086] 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.

[0087] 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.

[0088] 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.

[0089] Base candidate index defines the starting point. Base candidate index indicates the best candidate among candidates in the list as follows.

Table 2. Base candidate IDX

| Base candidate<br>IDX | 0                   | 1                   | 2                   | 3                   |
|-----------------------|---------------------|---------------------|---------------------|---------------------|
| N <sup>th</sup> MVP   | 1 <sup>st</sup> MVP | 2 <sup>nd</sup> MVP | 3 <sup>rd</sup> MVP | 4 <sup>th</sup> MVP |

[0090] If the number of base candidate is equal to 1, Base candidate IDX is not signaled.

[0091] 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 3. Distance IDX

| <b>Distance<br/>IDX</b>   | 0       | 1       | 2     | 3     | 4     | 5     | 6      | 7      |
|---------------------------|---------|---------|-------|-------|-------|-------|--------|--------|
| <b>Pixel<br/>distance</b> | 1/4-pel | 1/2-pel | 1-pel | 2-pel | 4-pel | 8-pel | 16-pel | 32-pel |

[0092] 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 4. Direction IDX

| <b>Direction IDX</b> | 00  | 01  | 10  | 11  |
|----------------------|-----|-----|-----|-----|
| <b>x-axis</b>        | +   | –   | N/A | N/A |
| <b>y-axis</b>        | N/A | N/A | +   | –   |

[0093] 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.

[0094] 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. 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.

[0095] UMVE is also known as Merge with MVD (MMVD).

[0096] FIG. 8 shows an example of an ultimate vector expression (UMVE) search process.

[0097] FIG. 9 shows an example of a UMVE search point.

## 2.5. Intra block copy

[0098] Intra block copy (IBC), a.k.a. current picture referencing, has been adopted in HEVC Screen Content Coding extensions (HEVC-SCC) and the current VVC test model (VTM-3.0). IBC

extends the concept of motion compensation from inter-frame coding to intra-frame coding. As demonstrated in FIG. 10, the current block is predicted by a reference block in the same picture when IBC is applied. The samples in the reference block have been already reconstructed before the current block is coded or decoded. Although IBC is not so efficient for most camera-captured sequences, it shows significant coding gains for screen content. The reason is that there are lots of repeating patterns, such as icons and text characters in a screen content picture. IBC can remove the redundancy between these repeating patterns effectively. In HEVC-SCC, an inter-coded coding unit (CU) can apply IBC if it chooses the current picture as its reference picture. The MV is renamed as block vector (BV) in this case, and a BV always has an integer-pixel precision. To be compatible with main profile HEVC, the current picture is marked as a “long-term” reference picture in the Decoded Picture Buffer (DPB). It should be noted that similarly, in multiple view/3D video coding standards, the inter-view reference picture is also marked as a “long-term” reference picture.

**[0099]** Following a BV to find its reference block, the prediction can be generated by copying the reference block. The residual can be got by subtracting the reference pixels from the original signals. Then transform and quantization can be applied as in other coding modes.

**[00100]** However, when a reference block is outside of the picture, or overlaps with the current block, or outside of the reconstructed area, or outside of the valid area restricted by some constraints, part or all pixel values are not defined. Basically, there are two solutions to handle such a problem. One is to disallow such a situation, e.g. in bitstream conformance. The other is to apply padding for those undefined pixel values. The following sub-sessions describe the solutions in detail.

## **2.6. IBC in HEVC Screen Content Coding extensions**

**[00101]** In the screen content coding extensions of HEVC, when a block uses the current picture as reference, it should guarantee that the whole reference block is within the available reconstructed area, as indicated in the following text from the specification:

The variables `offsetX` and `offsetY` are derived as follows:

$$\text{offsetX} = (\text{ChromaArrayType} == 0) ? 0 : (\text{mvCLX}[0] \& 0x7 ? 2 : 0) \quad (8-104)$$

$$\text{offsetY} = (\text{ChromaArrayType} == 0) ? 0 : (\text{mvCLX}[1] \& 0x7 ? 2 : 0) \quad (8-105)$$

It is a requirement of bitstream conformance that when the reference picture is the current picture, the luma motion vector `mvLX` shall obey the following constraints:

- When the derivation process for z-scan order block availability as specified in clause 6.4.1 is invoked with `(xCurr, yCurr)` set equal to `(xCb, yCb)` and the neighbouring luma location `(xNbY, yNbY)` set equal to `(xPb + (mvLX[0] >> 2) - offsetX, yPb + (mvLX[1] >> 2) - offsetY)` as inputs, the output shall be equal to TRUE.
- When the derivation process for z-scan order block availability as specified in clause 6.4.1 is invoked with `(xCurr, yCurr)` set equal to `(xCb, yCb)` and the neighbouring luma location `(xNbY, yNbY)` set equal to `(xPb + (mvLX[0] >> 2) + nPbW - 1 + offsetX, yPb + (mvLX[1] >> 2) + nPbH - 1 + offsetY)` as inputs, the output shall be equal to TRUE.
- One or both of the following conditions shall be true:
  - The value of `(mvLX[0] >> 2) + nPbW + xB1 + offsetX` is less than or equal to 0.
  - The value of `(mvLX[1] >> 2) + nPbH + yB1 + offsetY` is less than or equal to 0.
- The following condition shall be true:

$$(\text{xPb} + (\text{mvLX}[0] \gg 2) + \text{nPbSw} - 1 + \text{offsetX}) / \text{CtbSizeY} - \text{xCb} / \text{CtbSizeY} \leq \text{yCb} / \text{CtbSizeY} - (\text{yPb} + (\text{mvLX}[1] \gg 2) + \text{nPbSh} - 1 + \text{offsetY}) / \text{CtbSizeY} \quad (8-106)$$

[00102] Thus, the case that the reference block overlaps with the current block or the reference block is outside of the picture will not happen. There is no need to pad the reference or prediction block.

## 2.7. IBC in VVC Test Model

### 2.7.1. IBC Merge mode

[00103] In IBC merge mode, an index pointing to an entry in the IBC merge candidates list is parsed from the bitstream. The construction of the IBC merge list can be summarized according to the following sequence of steps:

- Step 1: Derivation of spatial candidates
- Step 2: Insertion of HMVP candidates
- Step 3: Insertion of pairwise average candidates

**[00104]** In the derivation of spatial merge candidates, a maximum of four merge candidates are selected among candidates located in the positions depicted in FIG. 11. 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 not coded with IBC mode. After candidate at position A1 is added, the insertion 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. 12 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.

**[00105]** After insertion of the spatial candidates, if the IBC merge list size is still smaller than the maximum IBC merge list size, IBC candidates from HMVP table may be inserted. Redundancy check are performed when inserting the HMVP candidates.

**[00106]** Finally, pairwise average candidates are inserted into the IBC merge list.

**[00107]** When a reference block identified by a merge candidate is outside of the picture, or overlaps with the current block, or outside of the reconstructed area, or outside of the valid area restricted by some constraints, the merge candidate is called invalid merge candidate.

**[00108]** It is noted that invalid merge candidates may be inserted into the IBC merge list.

### **2.7.2. IBC AMVP mode**

**[00109]** In IBC AMVP mode, an AMVP index point to an entry in the IBC AMVP list is parsed from the bitstream. The construction of the IBC AMVP list can be summarized according to the following sequence of steps:

- Step 1: Derivation of spatial candidates
  - Check A<sub>0</sub>, A<sub>1</sub> until an available candidate is found.
  - Check B<sub>0</sub>, B<sub>1</sub>, B<sub>2</sub> until an available candidate is found.
- Step 2: Insertion of HMVP candidates
- Step 3: Insertion of zero candidates

[00110] After insertion of the spatial candidates, if the IBC AMVP list size is still smaller than the maximum IBC AMVP list size, IBC candidates from HMVP table may be inserted.

[00111] Finally, zero candidates are inserted into the IBC AMVP list.

### 2.7.3. Allowed reference area in IBC

[00112] CPR block size is constrained to be no larger than 64x64 luma samples.

[00113] Each of the 64x64 blocks in the reference memory buffer is considered as whole, when some part of the 64x64 block has been updated with reconstructed samples from the current CTU, the reference samples from the left CTU in this whole 64x64 block cannot be used, as shown in FIG. 13 and FIG. 14. More specifically, depending on the location of the current coding block relative to the current CTU, the follows apply:

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

- If current block falls into the bottom-right 64x64 block of the current CTU, it can only refer to the already reconstructed samples in the current CTU, using CPR mode.

## **2.8. Adaptive motion vector difference resolution**

**[00114]** In HEVC, motion vector differences (MVDs) (between the motion vector and predicted motion vector of a PU) are signaled in units of quarter luma samples when `use_integer_mv_flag` is equal to 0 in the slice header. In the VVC test model (VTM), a locally adaptive motion vector resolution (AMVR) is introduced wherein the motion vector and motion vector differences are in the same precision, and each inter coded coding unit may select which resolution of one set of allowed precisions is utilized.

### **2.8.1. AMVR for translational inter mode**

**[00115]** In the VTM, MVD can be coded in units of quarter luma samples, integer luma samples or four luma samples. The MVD resolution is controlled at the coding unit (CU) level, and MVD resolution flags are conditionally signaled for each CU that has at least one non-zero MVD components.

**[00116]** For a CU that has at least one non-zero MVD components, a first flag is signaled 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 signaled to indicate whether integer luma sample MV precision or four luma sample MV precision is used.

**[00117]** 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.

**[00118]** In IBC inter mode, 1/4-pel precision is disabled and only 1-pel and 4-pel precision are allowed.

### 2.8.2. AMVR for affine inter mode

[00119] AMVR is extended to affine inter mode and MV precision set  $\{1/16, 1/4, 1\}$ -pel is used. If one MV precision is selected for an affine-coded block, both MVP candidates and MVD are rounded to the selected MV precision, as AMVR does.

[00120] At the encoder side, one of the three MV precisions could be selected for an affine-coded inter block. In VTM, AMVP mode and affine inter mode are checked together in one loop. For 1/4-pel MV precision, both AMVP mode and affine inter mode are checked. However, for 1-pel MV precision and 4-pel MV precision, only AMVP mode is checked. After extending AMVR to affine inter mode, affine inter mode using 1/16-pel MV precision is checked together with AMVP mode using 1-pel MV precision, and affine inter mode using 1-pel MV precision is checked together with AMVP mode using 4-pel MV precision.

[00121] Testing all three MV/MVD precisions in affine inter mode will significantly increase the encoding complexity. To address this issue, following fast algorithms is used:

1. If affine inter mode is not selected after checking rate-distortion costs of affine merge/skip mode, merge/skip mode, 1/4-pel MV precision AMVP mode and affine inter mode, then 1/16-pel MV precision and 1-pel MV precision affine inter modes are not checked.
2. Affine parameters obtained in 1/4-pel MV precision affine inter mode is reused as starting search point in 1/16-pel and 1-pel MV precision affine inter modes.

### 2.8.3. Syntax and semantics

#### 7.3.2.1 Sequence parameter set RBST syntax

| seq_parameter_set_rbsp( ) {                    | Descriptor |
|------------------------------------------------|------------|
| <b>sps_seq_parameter_set_id</b>                | ue(v)      |
| <b>intra_only_constraint_flag</b>              | u(1)       |
| <b>max_bitdepth_constraint_idc</b>             | u(4)       |
| <b>max_chroma_format_constraint_idc</b>        | u(2)       |
| <b>frame_only_constraint_flag</b>              | u(1)       |
| <b>no_qtbt_dual_tree_intra_constraint_flag</b> | u(1)       |
| <b>no_sao_constraint_flag</b>                  | u(1)       |
| <b>no_alf_constraint_flag</b>                  | u(1)       |
| <b>no_pcm_constraint_flag</b>                  | u(1)       |

|                                                                   |       |
|-------------------------------------------------------------------|-------|
| <b>no_temporal_mvp_constraint_flag</b>                            | u(1)  |
| <b>no_sbtmvp_constraint_flag</b>                                  | u(1)  |
| <b>no_amvr_constraint_flag</b>                                    | u(1)  |
| <b>no_cclm_constraint_flag</b>                                    | u(1)  |
| <b>no_affine_motion_constraint_flag</b>                           | u(1)  |
| <b>no_ladf_constraint_flag</b>                                    | u(1)  |
| <b>no_dep_quant_constraint_flag</b>                               | u(1)  |
| <b>no_sign_data_hiding_constraint_flag</b>                        | u(1)  |
| <b>chroma_format_idc</b>                                          | ue(v) |
| if( chroma_format_idc == 3 )                                      |       |
| <b>separate_colour_plane_flag</b>                                 | u(1)  |
| <b>pic_width_in_luma_samples</b>                                  | ue(v) |
| <b>pic_height_in_luma_samples</b>                                 | ue(v) |
| <b>bit_depth_luma_minus8</b>                                      | ue(v) |
| <b>bit_depth_chroma_minus8</b>                                    | ue(v) |
| <b>log2_max_pic_order_cnt_lsb_minus4</b>                          | ue(v) |
| <b>qtbtt_dual_tree_intra_flag</b>                                 | ue(v) |
| <b>log2_ctu_size_minus2</b>                                       | ue(v) |
| <b>log2_min_luma_coding_block_size_minus2</b>                     | ue(v) |
| <b>partition_constraints_override_enabled_flag</b>                | ue(v) |
| <b>sps_log2_diff_min_qt_min_cb_intra_tile_group_luma</b>          | ue(v) |
| <b>sps_log2_diff_min_qt_min_cb_inter_tile_group</b>               | ue(v) |
| <b>sps_max_mtt_hierarchy_depth_inter_tile_group</b>               | ue(v) |
| <b>sps_max_mtt_hierarchy_depth_intra_tile_group_luma</b>          | ue(v) |
| if( sps_max_mtt_hierarchy_depth_intra_tile_group_luma != 0 ) {    |       |
| <b>sps_log2_diff_max_bt_min_qt_intra_tile_group_luma</b>          | ue(v) |
| <b>sps_log2_diff_max_tt_min_qt_intra_tile_group_luma</b>          | ue(v) |
| }                                                                 |       |
| if( sps_max_mtt_hierarchy_depth_inter_tile_groups != 0 ) {        |       |
| <b>sps_log2_diff_max_bt_min_qt_inter_tile_group</b>               | ue(v) |
| <b>sps_log2_diff_max_tt_min_qt_inter_tile_group</b>               | ue(v) |
| }                                                                 |       |
| if( qtbtt_dual_tree_intra_flag ) {                                |       |
| <b>sps_log2_diff_min_qt_min_cb_intra_tile_group_chroma</b>        | ue(v) |
| <b>sps_max_mtt_hierarchy_depth_intra_tile_group_chroma</b>        | ue(v) |
| if ( sps_max_mtt_hierarchy_depth_intra_tile_group_chroma != 0 ) { |       |
| <b>sps_log2_diff_max_bt_min_qt_intra_tile_group_chroma</b>        | ue(v) |
| <b>sps_log2_diff_max_tt_min_qt_intra_tile_group_chroma</b>        | ue(v) |
| }                                                                 |       |
| }                                                                 |       |

|                                                       |       |
|-------------------------------------------------------|-------|
| <b>sps_sao_enabled_flag</b>                           | u(1)  |
| <b>sps_alf_enabled_flag</b>                           | u(1)  |
| <b>sps_pcm_enabled_flag</b>                           | u(1)  |
| if( sps_pcm_enabled_flag ) {                          |       |
| <b>pcm_sample_bit_depth_luma_minus1</b>               | u(4)  |
| <b>pcm_sample_bit_depth_chroma_minus1</b>             | u(4)  |
| <b>log2_min_pcm_luma_coding_block_size_minus3</b>     | ue(v) |
| <b>log2_diff_max_min_pcm_luma_coding_block_size</b>   | ue(v) |
| <b>pcm_loop_filter_disabled_flag</b>                  | u(1)  |
| }                                                     |       |
| <b>sps_ref_wraparound_enabled_flag</b>                | u(1)  |
| if( sps_ref_wraparound_enabled_flag )                 |       |
| <b>sps_ref_wraparound_offset</b>                      | ue(v) |
| <b>sps_temporal_mvp_enabled_flag</b>                  | u(1)  |
| if( sps_temporal_mvp_enabled_flag )                   |       |
| <b>sps_sbtmvp_enabled_flag</b>                        | u(1)  |
| sps_amvr_enabled_flag                                 | u(1)  |
| sps_bdof_enabled_flag                                 | u(1)  |
| sps_affine_amvr_enabled_flag                          | u(1)  |
| <b>sps_dmvr_enabled_flag</b>                          | u(1)  |
| <b>sps_cclm_enabled_flag</b>                          | u(1)  |
| if( sps_cclm_enabled_flag && chroma_format_idc == 1 ) |       |
| <b>sps_cclm_colocated_chroma_flag</b>                 | u(1)  |
| <b>sps_mts_enabled_flag</b>                           | u(1)  |
| if( sps_mts_enabled_flag ) {                          |       |
| <b>sps_explicit_mts_intra_enabled_flag</b>            | u(1)  |
| <b>sps_explicit_mts_inter_enabled_flag</b>            | u(1)  |
| }                                                     |       |
| ...                                                   |       |
| rbsp_trailing_bits()                                  |       |
| }                                                     |       |

#### 7.3.4.5 Coding unit syntax

|                                                      |                   |
|------------------------------------------------------|-------------------|
| coding_unit( x0, y0, cbWidth, cbHeight, treeType ) { | <b>Descriptor</b> |
| if( tile_group_type != I ) {                         |                   |
| if( treeType != DUAL_TREE_CHROMA )                   |                   |
| <b>cu_skip_flag</b> [ x0 ][ y0 ]                     | ae(v)             |
| if( cu_skip_flag[ x0 ][ y0 ] == 0 )                  |                   |
| <b>pred_mode_flag</b>                                | ae(v)             |
| }                                                    |                   |
| if( CuPredMode[ x0 ][ y0 ] == MODE_INTRA ) {         |                   |

|                                                                    |       |
|--------------------------------------------------------------------|-------|
| ...                                                                |       |
| }                                                                  |       |
| } else if( treeType != DUAL_TREE_CHROMA ) { /* MODE_INTER */       |       |
| if( cu_skip_flag[ x0 ][ y0 ] == 0 )                                |       |
| <b>merge_flag</b> [ x0 ][ y0 ]                                     | ae(v) |
| if( merge_flag[ x0 ][ y0 ] ) {                                     |       |
| merge_data( x0, y0, cbWidth, cbHeight )                            |       |
| } else {                                                           |       |
| if( tile_group_type == B )                                         |       |
| <b>inter_pred_idc</b> [ x0 ][ y0 ]                                 | ae(v) |
| if( sps_affine_enabled_flag && cbWidth >= 16 && cbHeight >= 16 ) { |       |
| <b>inter_affine_flag</b> [ x0 ][ y0 ]                              | ae(v) |
| if( sps_affine_type_flag && inter_affine_flag[ x0 ][ y0 ] )        |       |
| <b>cu_affine_type_flag</b> [ x0 ][ y0 ]                            | ae(v) |
| }                                                                  |       |
| if( inter_pred_idc[ x0 ][ y0 ] != PRED_L1 ) {                      |       |
| if( num_ref_idx_l0_active_minus1 > 0 )                             |       |
| <b>ref_idx_l0</b> [ x0 ][ y0 ]                                     | ae(v) |
| mvd_coding( x0, y0, 0, 0 )                                         |       |
| if( MotionModelIdc[ x0 ][ y0 ] > 0 )                               |       |
| mvd_coding( x0, y0, 0, 1 )                                         |       |
| if( MotionModelIdc[ x0 ][ y0 ] > 1 )                               |       |
| mvd_coding( x0, y0, 0, 2 )                                         |       |
| <b>mvp_l0_flag</b> [ x0 ][ y0 ]                                    | ae(v) |
| } else {                                                           |       |
| MvdL0[ x0 ][ y0 ][ 0 ] = 0                                         |       |
| MvdL0[ x0 ][ y0 ][ 1 ] = 0                                         |       |
| }                                                                  |       |
| if( inter_pred_idc[ x0 ][ y0 ] != PRED_L0 ) {                      |       |
| if( num_ref_idx_l1_active_minus1 > 0 )                             |       |
| <b>ref_idx_l1</b> [ x0 ][ y0 ]                                     | ae(v) |
| if( mvd_l1_zero_flag && inter_pred_idc[ x0 ][ y0 ] == PRED_BI ) {  |       |
| MvdL1[ x0 ][ y0 ][ 0 ] = 0                                         |       |
| MvdL1[ x0 ][ y0 ][ 1 ] = 0                                         |       |
| MvdCpL1[ x0 ][ y0 ][ 0 ][ 0 ] = 0                                  |       |
| MvdCpL1[ x0 ][ y0 ][ 0 ][ 1 ] = 0                                  |       |
| MvdCpL1[ x0 ][ y0 ][ 1 ][ 0 ] = 0                                  |       |
| MvdCpL1[ x0 ][ y0 ][ 1 ][ 1 ] = 0                                  |       |
| MvdCpL1[ x0 ][ y0 ][ 2 ][ 0 ] = 0                                  |       |
| MvdCpL1[ x0 ][ y0 ][ 2 ][ 1 ] = 0                                  |       |
| } else {                                                           |       |
| mvd_coding( x0, y0, 1, 0 )                                         |       |
| if( MotionModelIdc[ x0 ][ y0 ] > 0 )                               |       |
| mvd_coding( x0, y0, 1, 1 )                                         |       |
| if( MotionModelIdc[ x0 ][ y0 ] > 1 )                               |       |
| mvd_coding( x0, y0, 1, 2 )                                         |       |
| <b>mvp_l1_flag</b> [ x0 ][ y0 ]                                    | ae(v) |
| } else {                                                           |       |
| MvdL1[ x0 ][ y0 ][ 0 ] = 0                                         |       |

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |       |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| MvdL1[ x0 ][ y0 ][ 1 ] = 0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |       |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |
| if( ( sps_amvr_enabled_flag && inter_affine_flag == 0 &&<br>( MvdL0[ x0 ][ y0 ][ 0 ] != 0    MvdL0[ x0 ][ y0 ][ 1 ] != 0   <br>MvdL1[ x0 ][ y0 ][ 0 ] != 0    MvdL1[ x0 ][ y0 ][ 1 ] != 0 ) )   <br>( sps_affine_amvr_enabled_flag && inter_affine_flag[ x0 ][ y0 ] == 1 &&<br>( MvdCpL0[ x0 ][ y0 ][ 0 ][ 0 ] != 0    MvdCpL0[ x0 ][ y0 ][ 0 ][ 1 ] != 0   <br>MvdCpL1[ x0 ][ y0 ][ 0 ][ 0 ] != 0    MvdCpL1[ x0 ][ y0 ][ 0 ][ 1 ] != 0   <br>MvdCpL0[ x0 ][ y0 ][ 1 ][ 0 ] != 0    MvdCpL0[ x0 ][ y0 ][ 1 ][ 1 ] != 0   <br>MvdCpL1[ x0 ][ y0 ][ 1 ][ 0 ] != 0    MvdCpL1[ x0 ][ y0 ][ 1 ][ 1 ] != 0   <br>MvdCpL0[ x0 ][ y0 ][ 2 ][ 0 ] != 0    MvdCpL0[ x0 ][ y0 ][ 2 ][ 1 ] != 0   <br>MvdCpL1[ x0 ][ y0 ][ 2 ][ 0 ] != 0    MvdCpL1[ x0 ][ y0 ][ 2 ][ 1 ] != 0 ) ) { |       |
| if( !sps_cpr_enabled_flag    !(inter_pred_idc[ x0 ][ y0 ] == PRED_L0 &&<br>ref_idx_l0[ x0 ][ y0 ] == num_ref_idx_l0_active_minus1 ) )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |       |
| amvr_flag[ x0 ][ y0 ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ae(v) |
| if( amvr_flag[ x0 ][ y0 ] )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |       |
| amvr_precision_flag[ x0 ][ y0 ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | ae(v) |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |
| if( sps_gbi_enabled_flag && inter_pred_idc[ x0 ][ y0 ] == PRED_BI &&<br>cbWidth * cbHeight >= 256 )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |       |
| gbi_idx[ x0 ][ y0 ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | ae(v) |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |
| if( !pcm_flag[ x0 ][ y0 ] ) {                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |
| if( CuPredMode[ x0 ][ y0 ] != MODE_INTRA && merge_flag[ x0 ][ y0 ] == 0 )                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |       |
| cu_cbf                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | ae(v) |
| if( cu_cbf ) {                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |       |
| ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |       |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |
| }                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |       |

[00122] **sps\_amvr\_enabled\_flag** equal to 1 specifies that adaptive motion vector difference resolution is used in motion vector coding. **amvr\_enabled\_flag** equal to 0 specifies that adaptive motion vector difference resolution is not used in motion vector coding.

[00123] **sps\_affine\_amvr\_enabled\_flag** equal to 1 specifies that adaptive motion vector difference resolution is used in motion vector coding of affine inter mode. **sps\_affine\_amvr\_enabled\_flag** equal to 0 specifies that adaptive motion vector difference resolution is not used in motion vector coding of affine inter mode.

[00124] **amvr\_flag[ x0 ][ y0 ]** specifies the resolution of motion vector difference. The array indices **x0**, **y0** specify the location ( **x0**, **y0** ) of the top-left luma sample of the considered coding block relative to the top-left luma sample of the picture. **amvr\_flag[ x0 ][ y0 ]** equal to 0 specifies that

the resolution of the motion vector difference is 1/4 of a luma sample.  $\text{amvr\_flag}[x0][y0]$  equal to 1 specifies that the resolution of the motion vector difference is further specified by  $\text{amvr\_precision\_flag}[x0][y0]$ .

**[00125]** When  $\text{amvr\_flag}[x0][y0]$  is not present, it is inferred as follows:

- If  $\text{sps\_cpr\_enabled\_flag}$  is equal to 1,  $\text{amvr\_flag}[x0][y0]$  is inferred to be equal to 1.
- Otherwise (  $\text{sps\_cpr\_enabled\_flag}$  is equal to 0 ),  $\text{amvr\_flag}[x0][y0]$  is inferred to be equal to 0.

**[00126]**  $\text{amvr\_precision\_flag}[x0][y0]$  equal to 0 specifies that the resolution of the motion vector difference is one integer luma sample if  $\text{inter\_affine\_flag}[x0][y0]$  is equal to 0, and 1/16 of a luma sample otherwise.  $\text{amvr\_precision\_flag}[x0][y0]$  equal to 1 specifies that the resolution of the motion vector difference is four luma samples if  $\text{inter\_affine\_flag}[x0][y0]$  is equal to 0, and one integer luma sample otherwise. The array indices  $x0, y0$  specify the location (  $x0, y0$  ) of the top-left luma sample of the considered coding block relative to the top-left luma sample of the picture.

**[00127]** When  $\text{amvr\_precision\_flag}[x0][y0]$  is not present, it is inferred to be equal to 0.

**[00128]** The motion vector differences are modified as follows:

- If  $\text{inter\_affine\_flag}[x0][y0]$  is equal to 0, the variable  $\text{MvShift}$  is derived and the variables  $\text{MvdL0}[x0][y0][0]$ ,  $\text{MvdL0}[x0][y0][1]$ ,  $\text{MvdL1}[x0][y0][0]$ ,  $\text{MvdL1}[x0][y0][1]$  are modified as follows:

$$\text{MvShift} = ( \text{amvr\_flag}[x0][y0] + \text{amvr\_precision\_flag}[x0][y0] ) \ll 1 \quad (7-80)$$

$$\text{MvdL0}[x0][y0][0] = \text{MvdL0}[x0][y0][0] \ll ( \text{MvShift} + 2 ) \quad (7-81)$$

$$\text{MvdL0}[x0][y0][1] = \text{MvdL0}[x0][y0][1] \ll ( \text{MvShift} + 2 ) \quad (7-82)$$

$$\text{MvdL1}[x0][y0][0] = \text{MvdL1}[x0][y0][0] \ll ( \text{MvShift} + 2 ) \quad (7-83)$$

$$\text{MvdL1}[x0][y0][1] = \text{MvdL1}[x0][y0][1] \ll ( \text{MvShift} + 2 ) \quad (7-84)$$

- Otherwise (  $\text{inter\_affine\_flag}[x0][y0]$  is equal to 1 ), the variable  $\text{MvShift}$  is derived and the variables  $\text{MvdCpL0}[x0][y0][0][0]$ ,  $\text{MvdCpL0}[x0][y0][0][1]$ ,  $\text{MvdCpL0}[x0][y0][1][0]$ ,  $\text{MvdCpL0}[x0][y0][1][1]$ ,  $\text{MvdCpL0}[x0][y0][2][0]$  and  $\text{MvdCpL0}[x0][y0][2][1]$  are modified as follows:

$$MvShift = amvr\_coarse\_precisoin\_flag[x0][y0] \quad ?$$

$$(amvr\_coarse\_precisoin\_flag[x0][y0] \ll 1) : (- (amvr\_flag[x0][y0] \ll 1)) \quad (7-85)$$

$$MvdCpL0[x0][y0][0][0] = MvdCpL0[x0][y0][0][0] \ll (MvShift + 2) \quad (7-86)$$

$$MvdCpL1[x0][y0][0][1] = MvdCpL1[x0][y0][0][1] \ll (MvShift + 2) \quad (7-87)$$

$$MvdCpL0[x0][y0][1][0] = MvdCpL0[x0][y0][1][0] \ll (MvShift + 2) \quad (7-88)$$

$$MvdCpL1[x0][y0][1][1] = MvdCpL1[x0][y0][1][1] \ll (MvShift + 2) \quad (7-89)$$

$$MvdCpL0[x0][y0][2][0] = MvdCpL0[x0][y0][2][0] \ll (MvShift + 2) \quad (7-90)$$

$$MvdCpL1[x0][y0][2][1] = MvdCpL1[x0][y0][2][1] \ll (MvShift + 2) \quad (7-91)$$

### 3. Drawbacks of existing AMVR implementations

[00129] There are some potential problems in the current AMVR design for coding screen content:

- 1) Sub-pel motion vector is allowed in the normal inter mode although the chance to select sub-pel is relatively low.
- 2) For inter mode, 3 MV/MVD resolutions are allowed while for intra block copy/current picture referencing mode, only 2 MV/MVD resolutions are allowed which might be suboptimal.

### 4. Example embodiments of improved AMVR methods

[00130] To resolve above mentioned problems, several methods are described in this document to introduce modified Adaptive motion vector resolution (AMVR) design for screen content coding. Hereinafter, inter pictures that can only use current picture as the reference picture is referred as CPR only inter pictures.

[00131] The detailed embodiments described below should be considered as examples to explain general concepts. These embodiments should not be interpreted in a narrow way. Furthermore, these embodiments can be combined in any manner.

1. Sub-pel motion vector (MV) and/or motion vector predictor (MVP) and/or motion vector difference (MVD) may be disabled when the current tile/tile group/slice/picture belonging to screen content.

- a. Alternatively, furthermore, an indication of whether it is screen content may be signaled in tile/tile group/slice/picture/sequence/video parameter set or header.
  - b. Alternatively, furthermore, whether to disable or enable sub-pel MV and/or MVP and/or MVD may depend on the syntax for indication of enabling/disabling fractional MVD precisions in MMVD/UMVE mode.
  - c. Alternatively, furthermore, whether to disable or enable sub-pel MV and/or MVP and/or MVD may be signaled in tile/tile group/slice/picture/sequence/video parameter set or headers.
  - d. Motion vectors derived for each sub-block/block in affine mode may be rounded to integer-pel precision when the current tile/tile group/slice/picture belonging to screen content.
2. The set of allowed MV and/or MVP and/or MVD precisions in non-affine inter mode for current video/sequence/picture/tile group/slice may depend on one flag signaled in VPS/SPS/PPS/tile group header/slice header.
- a. In one example, the flag may be signaled in VPS/SPS/PPS/tile group header/slice header to indicate whether the current video/sequence/picture/tile group/slice is a screen content or not.
  - b. In one example, the flag indicating (e.g., the flag as described in bullet 5 of P1809117501H) whether fractional distance (or fractional MVD) is allowed or disallowed in MMVD for current video/sequence/picture/tile group/slice may be used.
  - c. In one example, if the flag is true (or false), {1/4, 1, 4}-pel MV and/or MVP and/or MVD precisions may be used.
  - d. In one example, if the flag is false (or true), {1, 2, 4}-pel MV and/or MVP and/or MVD precisions may be used.
    - i. In one example, {1, 2, 8}-pel MV and/or MVP and/or MVD precisions may be used.
    - ii. In one example, {1, 2, 16}-pel MV and/or MVP and/or MVD precisions may be used.
    - iii. In one example, {1, 4, 8}-pel MV and/or MVP and/or MVD precisions may be used.

- iv. In one example, {1, 4, 16}-pel MV and/or MVP and/or MVD precisions may be used.
  - v. In one example, {1, 8, 16}-pel MV and/or MVP and/or MVD precisions may be used.
  - vi. In one example, {1, 2}-pel MV and/or MVP and/or MVD precisions may be used.
  - vii. In one example, {1, 4}-pel MV and/or MVP and/or MVD precisions may be used.
  - viii. In one example, {1, 8}-pel MV and/or MVP and/or MVD precisions may be used.
  - ix. In one example, {1, 16}-pel MV and/or MVP and/or MVD precisions may be used.
3. The set of allowed MV and/or MVP and/or MVD precisions in affine inter mode for current video/sequence/picture/tile group/slice may depend on one flag signaled in VPS/SPS/PPS/tile group header/slice header.
- a. In one example, the flag may be signaled in VPS/SPS/PPS/sequence header/picture header/tile group header/slice header to indicate whether the current video/sequence/picture/tile group/slice is a screen content or not.
  - b. In one example, the flag indicating (as described in bullet 5 of P1809117501H) whether fractional distance (or fractional MVD) is allowed or disallowed in MMVD for current video/sequence/picture/tile group/slice may be used.
  - c. In one example, if the flag is true (or false), {1/4, 1, 1/16}-pel MV and/or MVP and/or MVD precisions may be used.
  - d. In one example, if the flag is false (or true), {1, 4, 1/4}-pel MV and/or MVP and/or MVD precisions may be used.
    - i. In one example, {1/2, 2, 1/8}-pel MV and/or MVP and/or MVD precisions may be used.
    - ii. In one example, {2, 8, 1/2}-pel MV and/or MVP and/or MVD precisions may be used.

- iii. In one example, {1, 2, 4}-pel MV and/or MVP and/or MVD precisions may be used.
  - iv. In one example, {1, 2, 8}-pel MV and/or MVP and/or MVD precisions may be used.
  - v. In one example, {1, 2, 16}-pel MV and/or MVP and/or MVD precisions may be used.
  - vi. In one example, {1, 4, 8}-pel MV and/or MVP and/or MVD precisions may be used.
  - vii. In one example, {1, 4, 16}-pel MV and/or MVP and/or MVD precisions may be used.
  - viii. In one example, {1, 8, 16}-pel MV and/or MVP and/or MVD precisions may be used.
  - ix. In one example, {1, 1/4}-pel MV and/or MVP and/or MVD precisions may be used.
  - x. In one example, {1, 1/2}-pel MV and/or MVP and/or MVD precisions may be used.
  - xi. In one example, {1, 1/8}-pel MV and/or MVP and/or MVD precisions may be used.
  - xii. In one example, {1, 1/16}-pel MV and/or MVP and/or MVD precisions may be used.
  - xiii. In one example, {1, 2}-pel MV and/or MVP and/or MVD precisions may be used.
  - xiv. In one example, {1, 4}-pel MV and/or MVP and/or MVD precisions may be used.
  - xv. In one example, {1, 8}-pel MV and/or MVP and/or MVD precisions may be used.
  - xvi. In one example, {1, 16}-pel MV and/or MVP and/or MVD precisions may be used.
4. Instead of using {1, 4}-pel MV and MVD precisions for IBC mode, it is proposed to use other kinds of two allowed MVD precisions.

- a. In one example, {1, 2}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
  - b. In one example, {1, 8}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
  - c. In one example, {1, 16}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
5. Same number of allowed MV and/or MVP and/or MVD resolutions for IBC mode may be used as that for normal inter mode.
  - a. As in current AMVR for normal inter mode, three MV and/or MVP and/or MVD precisions may be used.
    - i. In one example, {1, 2, 4}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
    - ii. In one example, {1, 2, 8}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
    - iii. In one example, {1, 2, 16}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
    - iv. In one example, {1, 4, 8}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
    - v. In one example, {1, 4, 16}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
    - vi. In one example, {1, 8, 16}-pel MV and/or MVP and/or MVD precisions may be used in IBC mode.
  - b. Alternatively, furthermore, same set of allowed MV and/or MVP and/or MVD resolutions for IBC mode may be used as that for normal inter mode, such as those mentioned above.
6. It is proposed that the semantics of the **amvr\_flag** in the specification is dependent on some high-level syntax such as those mentioned above.
  - a. In one example, amvr\_flag equal to 0 specifies that the resolution of the motion vector difference is 1/4 of a luma sample when the high-level syntax indicates sub-pel or fractional precision is allowed; and amvr\_flag equal to 1 specifies that the resolution

of the motion vector difference is  $M$  luma samples when the high-level syntax indicates sub-pel or fractional precision is disallowed.  $M$  is an integer value.

- i. In one example,  $M$  may be set to 1, or 2 or 4 or 8 or 16.
  - ii. In one example,  $M$  may be pre-defined or signaled in VPS/SPS/PPS/sequence header/picture header/tile group header/slice header.
  - iii. In one example,  $M$  may be conditionally signaled, such as based on the high-level syntax mentioned in 6.a.
  - iv. In one example,  $M$  may depend on the coded mode, such as normal inter mode and/or affine inter mode and/or IBC mode.
7. It is proposed that the semantics of the **amvr\_precision\_flag** in the specification is dependent on some high-level syntax such as those mentioned above.
  - a. In one example, when the high-level syntax indicates sub-pel or fractional precision is allowed, the current semantics is kept unchanged. However, when the high-level syntax indicates sub-pel or fractional precision is disallowed, the following may apply:
    - i. **amvr\_precision\_flag** equal to 0 specifies that the resolution of the motion vector difference is  $L0$  luma sample if **inter\_affine\_flag** is equal to 0, and  $L1$  of a luma sample otherwise. **amvr\_precision\_flag** equal to 1 specifies that the resolution of the motion vector difference is  $L2$  luma samples if **inter\_affine\_flag** is equal to 0, and  $L3$  luma sample otherwise.
  - b. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may be set to 1, or 2 or 4 or 8 or 16 or 1/16, 1/4, 1/2.
  - c. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may be pre-defined or signaled in VPS/SPS/PPS/sequence header/picture header/tile group header/slice header.
  - d. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may be conditionally signaled, such as based on the high-level syntax mentioned in 7.a.
  - e. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may depend on the coded mode, such as normal inter mode and/or affine inter mode and/or IBC mode.
8. In one example, the set of allowed MV and/or MVP and/or MVD precisions in non-affine or/and affine inter mode may be the same for screen content and none screen content, however,

semantics of **amvr\_flag** or/and **amvr\_precision\_flag** may depend on one flag signaled in VPS/SPS/PPS/tile group header/slice header, such as the flag described in bullet 1.

- a. In one example, if such flag indicates that current video/sequence/picture/tile group/slice is not screen content, semantics of **amvr\_flag** or/and **amvr\_precision\_flag** are kept unchanged (i.e., same as in VVC).
- b. In one example, if such flag indicates that current video/sequence/picture/tile group/slice is screen content, **amvr\_flag** equal to 0 specifies that the resolution of the motion vector difference is  $M$  luma samples.
  - i. In one example,  $M$  may be set equal to 1.
  - ii. In one example,  $M$  may be set equal to 2 or 4 or 8 or 16 or 1/16 or 1/4 or 1/2.
  - iii. In one example,  $M$  may be pre-defined or signaled in VPS/SPS/PPS/sequence header/picture header/tile group header/slice header.
  - iv. In one example,  $M$  may depend on the coded mode, such as normal inter mode and/or affine inter mode and/or IBC mode.
    1. For example,  $M$  may be set equal to 1 for normal inter mode and IBC mode, and set equal to 1/4 for affine inter mode.
    2. For example,  $M$  may be set equal to 1 for normal inter mode, IBC mode, and affine inter mode.
- c. In one example, if such flag indicates that current video/sequence/picture/tile group/slice is screen content, **amvr\_precision\_flag** equal to 0 specifies that the resolution of the motion vector difference is  $L0$  luma sample if **inter\_affine\_flag** is equal to 0, and  $L1$  of a luma sample otherwise. **amvr\_precision\_flag** equal to 1 specifies that the resolution of the motion vector difference is  $L2$  luma samples if **inter\_affine\_flag** is equal to 0, and  $L3$  luma sample otherwise.
  - i. In one example,  $L0$  may be set equal to 1/4,  $L1$  may be set equal to 1/4,  $L2$  may be set equal to 4 and  $L3$  may be set equal to 1/16.
  - ii. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may be set to 1, or 2 or 4 or 8 or 16 or 1/16, 1/4, 1/2.
  - iii. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may be pre-defined or signaled in VPS/SPS/PPS/sequence header/picture header/tile group header/slice header.

- iv. In one example, any of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  may depend on the coded mode, such as normal inter mode and/or affine inter mode and/or IBC mode.

## 5. Example implementations of the disclosed technology

[00132] FIG. 15 is a block diagram of a video processing apparatus 1500. The apparatus 1500 may be used to implement one or more of the methods described herein. The apparatus 1500 may be embodied in a smartphone, tablet, computer, Internet of Things (IoT) receiver, and so on. The apparatus 1500 may include one or more processors 1502, one or more memories 1504 and video processing hardware 1506. The processor(s) 1502 may be configured to implement one or more methods described in the present document. The memory (memories) 1504 may be used for storing data and code used for implementing the methods and techniques described herein. The video processing hardware 1506 may be used to implement, in hardware circuitry, some techniques described in the present document, and may be partly or completely be a part of the processors 1502 (e.g., graphics processor core GPU or other signal processing circuitry).

[00133] In the present document, the term “video processing” may refer to video encoding, video decoding, video compression or video decompression. For example, video compression algorithms may be applied during conversion from pixel representation of a video to a corresponding bitstream representation or vice versa. The bitstream representation of a current video block may, for example, correspond to bits that are either co-located or spread in different places within the bitstream, as is defined by the syntax. For example, a macroblock may be encoded in terms of transformed and coded error residual values and also using bits in headers and other fields in the bitstream.

[00134] It will be appreciated that several techniques have been disclosed that will benefit video encoder and decoder embodiments incorporated within video processing devices such as smartphones, laptops, desktops, and similar devices by allowing the use of the techniques disclosed in the present document.

[00135] FIG. 16 is a flowchart for an example method 1600 of video processing. The method 1600 includes, at 1610, determining that a member of the group including (i) a current tile, (ii) a current tile group, (iii) a current slice, or (iv) a current picture is associated with on-screen displayed content. The method 1600 also includes, at 1620, in response to the determining, enabling or disabling one or more sub-pixel motion vector parameters.

[00136] FIG. 17 is a flowchart for an example method 1600 of video processing. The method 1700 includes, at 1710, determining, based on a first indication in a bitstream representation of a current video block, whether a subpel motion precision for motion information is allowed for the current video block, wherein motion information comprises at least one of a motion vector (MV), a motion vector prediction (MVP), and a motion vector difference (MVD). The method 1700 also includes, at 1720, performing, a conversion between the current video block and the bitstream representation of the current video block, using a non-subpel motion precision for the motion information after determining that the subpel motion precision is disallowed.

[00137] FIG. 18 is a flowchart for an example method 1600 of video processing. The method 1800 includes, at 1810, determining, during a conversion between a current video block and a bitstream representation of the current video block, a set of allowed motion precisions for motion information, wherein the current video block is coded in intra block copy (IBC) mode, wherein the motion information comprises at least one of a motion vector, a motion vector prediction and a motion vector difference. The method 1800 also includes, at 1820, performing the conversion, based on the determined set of motion precisions.

[00138] Some embodiments may be described using the following clause-based format.

[00139] 1. A method for video processing, comprising:

[00140] determining that a member of the group including (i) a current tile, (ii) a current tile group, (iii) a current slice, or (iv) a current picture is associated with on-screen displayed content; and

[00141] in response to the determining, enabling or disabling one or more sub-pixel motion vector parameters.

[00142] 2. The method of clause 1, wherein the sub-pixel motion vector parameters include any of: a motion vector, a motion vector predictor, or a motion vector difference.

[00143] 3. The method of clause 1, wherein an indication of the on-screen displayed content is signaled in a tile, a tile group, a slice, a picture, a sequence, or a video parameter set.

[00144] 4. The method of clause 1, wherein an indication of the on-screen displayed content is signaled in a tile, a tile group, a slice, a picture, a sequence, or a video parameter header.

- [00145] 5. The method of clause 1, wherein enabling or disabling the one or more sub-pixel motion vector parameters is associated with a syntax of fractional motion vector difference precision.
- [00146] 6. The method of clause 1, wherein enabling or disabling the one or more sub-pixel motion vector parameters is signaled in a tile, a tile group, a slice, a picture, a sequence, or a video parameter set.
- [00147] 7. The method of clause 1, wherein enabling or disabling the one or more sub-pixel motion vector parameters is signaled in a tile, a tile group, a slice, a picture, a sequence, or a video parameter header.
- [00148] 8. The method of clause 1, further comprising:
- [00149] in response to determining that a member of the group is associated with the on-screen displayed content, rounding the one or more sub-pixel motion vector parameters to one or more integer-pixel motion vector parameters.
- [00150] 9. The method of clause 8, wherein the one or more sub-pixel motion vector parameters are for each block or sub-block, and wherein an affine mode is set.
- [00151] 10. The method of clause 1, wherein a non-affine inter mode is set, and wherein allowed motion vector parameters depend on a value of a flag included in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00152] 11. The method of clause 10, wherein the value of the flag included in the VPS, the SPS, the PPS, the tile group header, the tile header, or the slice header indicates whether (i) the current tile, (ii) the current tile group, (iii) the current slice, or (iv) the current picture is associated with on-screen displayed content.
- [00153] 12. The method of clause 10, wherein the flag indicates whether a fractional distance or a fractional motion vector difference is allowed or disallowed.
- [00154] 13. The method of clause 10, wherein depending on the value of the flag, the motion vector parameters have pixel precisions according to one of the following sets:  $\{1/4, 1, 4\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8, 16\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ .

**[00155]** 14. The method of clause 1, wherein an affine inter mode is set, and wherein allowed motion vector parameters depend on a value of a flag included in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.

**[00156]** 15. The method of clause 14, wherein the value of the flag included in the VPS, the SPS, the PPS, the tile group header, the tile header, or the slice header indicates whether (i) the current tile, (ii) the current tile group, (iii) the current slice, or (iv) the current picture is associated with on-screen displayed content.

**[00157]** 16. The method of clause 15, wherein the flag indicates whether a fractional distance or a fractional motion vector difference is allowed or disallowed.

**[00158]** 17. The method of clause 16, wherein depending on the value of the flag, the motion vector parameters have pixel precisions according to one of the following sets:  $\{1/4, 1, 1/16\}$ ,  $\{1, 4, 1/4\}$ ,  $\{1/2, 2, 1/8\}$ ,  $\{2, 8, 1/2\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8, 16\}$ ,  $\{1, 1/4\}$ ,  $\{1, 1/2\}$ ,  $\{1, 1/8\}$ ,  $\{1, 1/16\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ .

**[00159]** 18. The method of clause 1, wherein an IBC mode is set, and wherein allowed motion vector parameters depend on a value of a flag included in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.

**[00160]** 19. The method of clause 18, wherein the value of the flag included in the VPS, the SPS, the PPS, the tile group header, the tile header, or the slice header indicates whether (i) the current tile, (ii) the current tile group, (iii) the current slice, or (iv) the current picture is associated with on-screen displayed content.

**[00161]** 20. The method of clause 19, wherein the flag indicates whether a fractional distance or a fractional motion vector difference is allowed or disallowed.

**[00162]** 21. The method of clause 16, wherein depending on the value of the flag, the motion vector parameters have pixel precisions according to one of the following sets:  $\{1, 2\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ .

- [00163] 22. The method of clause 1, wherein an allowed number of resolutions of the motion vector parameters is same for a normal inter mode and an IBC mode.
- [00164] 23. The method of clause 22, wherein the IBC mode is set, and wherein the motion vector parameters have pixel precisions according to one of the following sets: {1, 2, 4}, {1, 2, 8}, {1, 2, 16}, {1, 4, 8}, {1, 4, 16}, or {1, 8, 16}.
- [00165] 24. The method of clause 1, wherein a value of an *amvr\_flag* equals to 0 specifies that a resolution of the motion vector parameters is 1/4 of a luma sample when a high-level syntax indicates enabling the one or more sub-pixel motion vector parameters.
- [00166] 25. The method of clause 1, wherein a value of an *amvr\_flag* equals to 1 specifies that a resolution of the motion vector parameters is *M* luma samples when a high-level syntax indicates disabling the one or more sub-pixel motion vector parameters.
- [00167] 26. The method of clause 25, wherein *M* is 1, 2, 4, 8, or 16.
- [00168] 27. The method of clause 25, wherein *M* is specified in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00169] 28. The method of clause 25, wherein *M* depends on whether a normal inter mode, an affine inter mode, or an IBC mode is set.
- [00170] 29. The method of clause 1, wherein when the determining indicates disabling the one or more sub-pixel motion vector parameters, a value of an *amvr\_precision\_flag* equals to 0 specifies that a resolution of the motion vector parameters is *L0* luma samples if *inter\_affine\_flag* is equal to 0, and *L1* luma samples otherwise.
- [00171] 30. The method of clause 1, wherein when the determining indicates disabling the one or more sub-pixel motion vector parameters, a value of an *amvr\_precision\_flag* equals to 1 specifies that a resolution of the motion vector parameters is *L2* luma samples if *inter\_affine\_flag* is equal to 0, and *L3* luma samples otherwise.
- [00172] 31. The method of clauses 29 through 30, wherein *L0*, *L1*, *L2*, *L3* take values from the set {1, 2, 4, 8, 16, 1/16, 1/4, 1/2}.

- [00173] 32. The method of clauses 29 through 30, wherein  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  have predetermined values.
- [00174] 33. The method of clauses 29 through 30, wherein  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  take values specified in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00175] 34. The method of clause 29 through 30, wherein values of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  depend on whether a normal inter mode, an affine inter mode, or an IBC mode is set.
- [00176] 35. The method of clause 1, wherein an allowed number of resolutions of the motion vector parameters is same for an affine inter mode and a non-affine mode regardless of the determining.
- [00177] 36. The method of clause 35, wherein a value of an *amvr\_flag* and a value of an *amvr\_precision\_flag* is specified as a third flag in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00178] 37. The method of clause 36, wherein if the third flag indicates that a member of the group is not associated with on-screen displayed content, values of the *amvr\_flag* and the *amvr\_precision\_flag* are unchanged.
- [00179] 38. The method of clause 35, wherein if the third flag indicates that a member of the group is associated with on-screen displayed content, *amvr\_flag* is set to 0 specifying that a resolution of the motion vector parameters is  $M$  luma samples.
- [00180] 39. The method of clause 38, wherein  $M$  equals 1.
- [00181] 40. The method of clause 38, wherein  $M$  equals 2, 4, 8, 16, 1/16, 1/4, or 1/2.
- [00182] 41. The method of clause 38, wherein  $M$  is specified in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00183] 42. The method of clause 38, wherein  $M$  depends on whether a normal inter mode, an affine inter mode, or an IBC mode is set.

- [00184] 43. The method of clause 38, wherein if the third flag indicates that a member of the group is associated with on-screen displayed content, *amvr\_precision\_flag* is set to 0 specifying that a resolution of the motion vector parameters is *L0* luma samples if *inter\_affine\_flag* is equal to 0, and *L1* luma samples otherwise.
- [00185] 44. The method of clause 38, wherein if the third flag indicates that a member of the group is associated with on-screen displayed content, *amvr\_precision\_flag* is set to 1 specifying that a resolution of the motion vector parameters is *L2* luma samples if *inter\_affine\_flag* is equal to 0, and *L3* luma samples otherwise.
- [00186] 45. The method of clauses 43 through 44, wherein *L0*, *L1*, *L2*, *L3* take values from the set {1, 2, 4, 8, 16, 1/16, 1/4, 1/2}.
- [00187] 46. The method of clauses 43 through 44, wherein *L0*, *L1*, *L2*, *L3* have predetermined values.
- [00188] 47. The method of clauses 43 through 44, wherein *L0*, *L1*, *L2*, *L3* take values specified in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00189] 48. The method of clause 43 through 44, wherein values of *L0*, *L1*, *L2*, *L3* depend on whether a normal inter mode, an affine inter mode, or an IBC mode is set.
- [00190] 49. The method of clauses 43 through 44, wherein *L0* is set to 1/4, *L1* is set to 1/4, *L2* is set to 4, and *L3* is set to 1/16.
- [00191] 50. The method of any one or more of clauses 1 through 49, wherein the video processing is an encoder-side implementation.
- [00192] 51. The method of any one or more of clauses 1 through 49, wherein the video processing is a decoder-side implementation.
- [00193] 52. 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 or more of clauses 1 to 50.

[00194] 53. 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 or more of clauses 1 to 50.

[00195] Some embodiments may be described using the following clause-based format.

[00196] 1. A method for video processing, comprising:

[00197] determining, based on a first indication in a bitstream representation of a current video block, whether a subpel motion precision for motion information is allowed for the current video block, wherein motion information comprises at least one of a motion vector (MV), a motion vector prediction (MVP), and a motion vector difference (MVD);

[00198] performing, a conversion between the current video block and the bitstream representation of the current video block, using a non-subpel motion precision for the motion information after determining that the subpel motion precision is disallowed.

[00199] 2. The method of clause 1, wherein the first indication comprises a first syntax signaled in at least one of a tile, a tile group, a slice, a picture, a sequence, a video parameter set and a video parameter header.

[00200] 3. The method of clause 2, wherein the first syntax indicates whether a tile, a tile group, a slice, a picture, a sequence or a video belongs to a screen content.

[00201] 4. The method of anyone clause 1-3, wherein the subpel motion precision is disallowed for a current processing unit in response to the current processing unit belongs to a screen content, wherein the current processing unit comprises a tile, a tile group, a slice, a picture, a sequence or a video.

[00202] 5. The method of anyone of clause 1-4, wherein the first indication comprises a second syntax indicating whether the subpel motion precision is allowed in a merge mode with a motion vector difference (MMVD) mode.

[00203] 6. The method of clause 5, wherein the subpel motion precision is disallowed for the current video block in response to the subpel motion precision being disallowed in the MMVD mode.

**[00204]** 7 The method of anyone of clauses 1-6, wherein the first indication comprises a third syntax signaled in at least one of a tile, a tile group, a slice, a picture, a sequence, a video parameter set and a video parameter header to indicate whether the subpel motion precision is disallowed for the tile, the tile group, the slice, the picture, the sequence or the video.

**[00205]** 8 The method of anyone of clauses 1-7, wherein the motion information of the current video block is rounded to an integer-pixel precision in response to the subpel motion precision being disallowed for the current video block.

**[00206]** 9 The method of anyone of clauses 1-8, further comprising:

**[00207]** determining, based on a second indication in the bitstream representation of the current video block, a set of motion precisions for the motion information allowed for the current video block;

**[00208]** performing, the conversion between the current video block and the bitstream representation of the current video block, based the determined set of motion precisions.

**[00209]** 10 The method of clause 9, wherein the second indication and the first indication are realized by a same flag.

**[00210]** 11. The method of clause 9 or 10, wherein the set of motion precisions for a non-affine inter mode comprises  $\{1/4, 1, 4\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8, 16\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ -sample precision, in response to the second indication being true.

**[00211]** 12. The method of anyone of clauses 9-11, wherein the set of motion precisions for a non-affine inter mode comprises  $\{1/4, 1, 4\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8, 16\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ -sample precision, in response to the second indication being false.

**[00212]** 13. The method of anyone of clauses 9-12, wherein the set of motion precisions for an affine inter mode comprises  $\{1/4, 1, 1/16\}$ ,  $\{1, 4, 1/4\}$ ,  $\{1/2, 2, 1/8\}$ ,  $\{2, 8, 1/2\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8, 16\}$ ,  $\{1, 1/4\}$ ,  $\{1, 1/2\}$ ,  $\{1, 1/8\}$ ,  $\{1, 1/16\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ -sample precision, in response to the second indication being true.

**[00213]** 14. The method of anyone of clauses 9-13, wherein the set of motion precisions for an affine inter mode comprises  $\{1/4, 1, 1/16\}$ ,  $\{1, 4, 1/4\}$ ,  $\{1/2, 2, 1/8\}$ ,  $\{2, 8, 1/2\}$ ,  $\{1, 2, 4\}$ ,  $\{1,$

2, 8}, {1, 2, 16}, {1, 4, 8}, {1, 4, 16}, {1, 8, 16}, {1, 1/4}, {1, 1/2}, {1, 1/8}, {1, 1/16}, {1, 2}, {1, 4}, {1, 8}, or {1, 16}-sample precision, in response to the second indication being false.

**[00214]** 15 The method of anyone of clauses 1-14, further comprising:

**[00215]** determining, based on a third indication in the bitstream representation of the current video block, a default motion precision for the motion information for the current video block; and

**[00216]** determining, based on a fourth indication in a bitstream representation of the current video block, whether an adaptive motion vector resolution (AMVR) is enabled for the current video block; and

**[00217]** performing, the conversion between the current video block and the bitstream representation of the current video block, based on the default motion precision in response to the AMVR being disabled for the current video block.

**[00218]** 16. The method of clause 15, wherein the third indication and the first indication are realized by a same flag.

**[00219]** 17. The method of clause 15 or 16, wherein the fourth indication comprises a fourth syntax indicating whether the AMVR is enabled for the current video block.

**[00220]** 18. The method of anyone of clauses 15-17, wherein the fourth indication is `amvr_flag`, and `amvr_flag` with a value of 0 indicates that the AMVR is disabled for the current video block and `amvr_flag` with a value of 1 indicates that the AMVR is enabled for the current video block.

**[00221]** 19. The method of anyone of clauses 15-18, wherein the default motion precision is set equal to 1/4 luma sample precision when the third indication indicates that subpel motion precision is allowed.

**[00222]** 20. The method of anyone of clauses 15-19, wherein the default motion precision is set equal to  $M1$  luma sample precision when the third indication indicates that subpel motion precision is disallowed.

**[00223]** 21. The method of anyone of clauses 15-20, wherein the default motion precision is set equal to  $M2$  luma sample precision when the third indication indicates that the current video block is a screen content.

- [00224] 22. The method of clause 20, wherein  $M1$  is of a value from 1, 2, 4, 8, and 16.
- [00225] 23. The method of clause 21, wherein  $M2$  is of a value 1, 2, 4, 8, 16, 1/16, 1/4 or 1/2.
- [00226] 24. The method of anyone of clauses 21-23, wherein  $M1$  and  $M2$  is signaled in at least one of a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.
- [00227] 25. The method of anyone of clauses 21-24, wherein  $M1$  and  $M2$  is signaled based on the third indication.
- [00228] 26. The method of anyone of clauses 21-24, wherein  $M1$  and  $M2$  is based on a coded mode which is one of a non-affine inter mode, an affine inter mode and an IBC mode.
- [00229] 27. The method of anyone of clauses 1-26, further comprising:
- [00230] determining, based on a fifth indication and a sixth indication in the bitstream representation of the video, a non-default motion precision for motion information used for the current video block, wherein adaptive motion vector resolution (AMVR) is enabled for the current video block; and
- [00231] performing, the conversion between the current video block and the bitstream representation of the current video block, based on the determined motion precision.
- [00232] 28. The method of clause 27, wherein the fifth indication is same as the first indication.
- [00233] 29. The method of clause 27 or 28, wherein the sixth indication is an `amvr_precision_flag`.
- [00234] 30. The method of clause 29, wherein the `amvr_precision_flag` with a value of 0 indicates the motion precision is  $L0$  luma samples for a non-affine inter mode and  $L1$  luma samples for an affine inter mode, in response to the fifth indication indicates that subpel motion precision is disallowed.
- [00235] 31. The method of clause 29, wherein the `amvr_precision_flag` with a value of 1 indicates the motion precision is  $L2$  luma samples for a non-affine inter mode and  $L3$  luma samples

for an affine inter mode, in response to the fifth indication indicates that subpel motion precision is allowed.

[00236] 32. The method of clause 29, wherein the `amvr_precision_flag` with a value of 0 indicates the motion precision is *L0* luma samples for a non-affine inter mode and *L1* luma samples for an affine inter mode, in response to the fifth indication indicates that a current video block is not a screen content.

[00237] 33. The method of clause 29, wherein the `amvr_precision_flag` with a value of 1 indicates the motion precision is *L2* luma samples for a non-affine inter mode and *L3* luma samples for an affine inter mode, in response to the fifth indication indicates that a current video block is a screen content.

[00238] 34. The method of clause 33, wherein anyone of *L0*, *L1*, *L2*, *L3* is of a value from 1, 2, 4, 8, 16, 1/16, 1/4, 1/2.

[00239] 35. The method of clause 33, wherein anyone of *L0*, *L1*, *L2*, *L3* is signaled in at least one of a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header

[00240] 36. The method of clause 33, wherein anyone of *L0*, *L1*, *L2*, *L3* is signaled based on the fifth indication.

[00241] 37. The method of clause 33, wherein anyone of *L0*, *L1*, *L2*, *L3* is based on a coded mode which is one of a non-affine inter mode, an affine inter mode and an IBC mode.

[00242] 38. The method of anyone of clauses 1-37, wherein a first set of allowed motion precisions for a non-affine inter mode of the screen content is same with a second set of allowed motion precisions for a non-affine inter mode of a none screen content.

[00243] 39. The method of anyone of clauses 1-37, wherein a first set of allowed motion precisions for an affine inter mode of the screen content is same with a second set of allowed motion precisions for an affine inter mode of a none screen content.

[00244]

[00245] 40. A video processing apparatus comprising a processor configured to implement a method recited in anyone of clauses 1 to 39.

- [00246] 41. The apparatus of clause 40, wherein the apparatus is a video encoder.
- [00247] 42. The apparatus of clause 40, wherein the apparatus is a video decoder.
- [00248] 43. A computer readable recoding media on which a program comprising code is recorded, the program is for a processor to carry out a method recited in anyone of clauses 1 to 39.
- [00249] Some embodiments may be described using the following clause-based format.
- [00250] 1. A method for video processing, comprising:
- [00251] determining, during a conversion between a current video block and a bitstream representation of the current video block, a set of allowed motion precisions for motion information, wherein the current video block is coded in intra block copy (IBC) mode, wherein the motion information comprises at least one of a motion vector, a motion vector prediction and a motion vector difference;
- [00252] performing the conversion, based on the determined set of motion precisions.
- [00253] 2. The method of clause 1, wherein the determined set of allowed motion precisions comprises {1, 2}-sample precision.
- [00254] 3. The method of clause 1, wherein the determined set of allowed motion precisions comprises {1, 8}-sample precision.
- [00255] 4. The method of clause 1, wherein the determined set of allowed motion precisions comprises {1, 16}-sample precision.
- [00256] 5. The method of anyone of clauses 1-4, wherein a first number of the allowed motion precisions for an IBC mode is same as a second number of the allowed motion precisions for a normal inter mode .
- [00257] 6. The method of clause 5, wherein the normal inter mode comprises a non-affine advanced motion vector prediction (AMVP) mode.
- [00258] 7. The method of clause 5, wherein the second number is 3.
- [00259] 8. The method of anyone of clauses 1-7, the set of allowed MV/MVP/MVD precisions for an IBC mode comprises {1, 2, 4}, {1, 2, 8}, {1, 2, 16}, {1, 4, 8}, {1, 4, 16} or {1, 8, 16 }-sample precision.

[00260] 9. The method of anyone of clauses 1-8, wherein the set of the allowed motion precisions for an IBC mode is same as a set of the allowed motion precisions for a normal inter mode .

[00261] 10. A video processing apparatus comprising a processor configured to implement a method recited in anyone of clauses 1 to 9.

[00262] 11. The apparatus of clause 10, wherein the apparatus is a video encoder.

[00263] 12. The apparatus of clause 10, wherein the apparatus is a video decoder.

[00264] 13. A computer readable recoding media on which a program comprising code is recorded, the program is for a processor to carry out a method recited in anyone of clauses 1 to 9.

[00265] 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.

[00266] 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.

**[00267]** 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).

**[00268]** 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.

**[00269]** 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.

**[00270]** 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.

**[00271]** 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 for video processing, comprising:

determining, based on a first indication in a bitstream representation of a current video block, whether a subpel motion precision for motion information is allowed for the current video block, wherein motion information comprises at least one of a motion vector (MV), a motion vector prediction (MVP), and a motion vector difference (MVD);

performing, a conversion between the current video block and the bitstream representation of the current video block, using a non-subpel motion precision for the motion information after determining that the subpel motion precision is disallowed.

2. The method of claim 1, wherein the first indication comprises a first syntax signaled in at least one of a tile, a tile group, a slice, a picture, a sequence, a video parameter set and a video parameter header.

3. The method of claim 2, wherein the first syntax indicates whether a tile, a tile group, a slice, a picture, a sequence or a video belongs to a screen content.

4. The method of anyone claim 1-3, wherein the subpel motion precision is disallowed for a current processing unit in response to the current processing unit belongs to a screen content, wherein the current processing unit comprises a tile, a tile group, a slice, a picture, a sequence or a video.

5. The method of anyone of claim 1-4, wherein the first indication comprises a second syntax indicating whether the subpel motion precision is allowed in a merge mode with a motion vector difference (MMVD) mode.

6. The method of claim 5, wherein the subpel motion precision is disallowed for the current video block in response to the subpel motion precision being disallowed in the MMVD mode.

7 The method of anyone of claims 1-6, wherein the first indication comprises a third syntax signaled in at least one of a tile, a tile group, a slice, a picture, a sequence, a video parameter set and a video parameter header to indicate whether the subpel motion precision is disallowed for the tile, the tile group, the slice, the picture, the sequence or the video.

8 The method of anyone of claims 1-7, wherein the motion information of the current video block is rounded to an integer-pixel precision in response to the subpel motion precision being disallowed for the current video block.

9 The method of anyone of claims 1-8, further comprising:

determining, based on a second indication in the bitstream representation of the current video block, a set of motion precisions for the motion information allowed for the current video block;

performing, the conversion between the current video block and the bitstream representation of the current video block, based the determined set of motion precisions.

10 The method of claim 9, wherein the second indication and the first indication are realized by a same flag.

11. The method of claim 9 or 10, wherein the set of motion precisions for a non-affine inter mode comprises  $\{1/4, 1, 4\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8, 16\}$ ,  $\{1, 2\}$ ,  $\{1, 4\}$ ,  $\{1, 8\}$ , or  $\{1, 16\}$ -sample precision, in response to the second indication being true.

12. The method of anyone of claims 9-11, wherein the set of motion precisions for a non-affine inter mode comprises  $\{1/4, 1, 4\}$ ,  $\{1, 2, 4\}$ ,  $\{1, 2, 8\}$ ,  $\{1, 2, 16\}$ ,  $\{1, 4, 8\}$ ,  $\{1, 4, 16\}$ ,  $\{1, 8,$

16}, {1, 2}, {1, 4}, {1, 8}, or {1, 16}-sample precision, in response to the second indication being false.

13. The method of anyone of claims 9-12, wherein the set of motion precisions for an affine inter mode comprises {1/4, 1, 1/16}, {1, 4, 1/4}, {1/2, 2, 1/8}, {2, 8, 1/2}, {1, 2, 4}, {1, 2, 8}, {1, 2, 16}, {1, 4, 8}, {1, 4, 16}, {1, 8, 16}, {1, 1/4}, {1, 1/2}, {1, 1/8}, {1, 1/16}, {1, 2}, {1, 4}, {1, 8}, or {1, 16}-sample precision, in response to the second indication being true.

14. The method of anyone of claims 9-13, wherein the set of motion precisions for an affine inter mode comprises {1/4, 1, 1/16}, {1, 4, 1/4}, {1/2, 2, 1/8}, {2, 8, 1/2}, {1, 2, 4}, {1, 2, 8}, {1, 2, 16}, {1, 4, 8}, {1, 4, 16}, {1, 8, 16}, {1, 1/4}, {1, 1/2}, {1, 1/8}, {1, 1/16}, {1, 2}, {1, 4}, {1, 8}, or {1, 16}-sample precision, in response to the second indication being false.

15 The method of anyone of claims 1-14, further comprising:

determining, based on a third indication in the bitstream representation of the current video block, a default motion precision for the motion information for the current video block; and

determining, based on a fourth indication in a bitstream representation of the current video block, whether an adaptive motion vector resolution (AMVR) is enabled for the current video block; and

performing, the conversion between the current video block and the bitstream representation of the current video block, based on the default motion precision in response to the AMVR being disabled for the current video block.

16. The method of claim 15, wherein the third indication and the first indication are realized by a same flag.

17. The method of claim 15 or 16, wherein the fourth indication comprises a fourth syntax indicating whether the AMVR is enabled for the current video block.

18. The method of anyone of claims 15-17, wherein the fourth indication is *amvr\_flag*, and *amvr\_flag* with a value of 0 indicates that the AMVR is disabled for the current video block and *amvr\_flag* with a value of 1 indicates that the AMVR is enabled for the current video block.

19. The method of anyone of claims 15-18, wherein the default motion precision is set equal to 1/4 luma sample precision when the third indication indicates that subpel motion precision is allowed.

20. The method of anyone of claims 15-19, wherein the default motion precision is set equal to *M1* luma sample precision when the third indication indicates that subpel motion precision is disallowed.

21. The method of anyone of claims 15-20, wherein the default motion precision is set equal to *M2* luma sample precision when the third indication indicates that the current video block is a screen content.

22. The method of claim 20, wherein *M1* is of a value from 1, 2, 4, 8, and 16.

23. The method of claim 21, wherein *M2* is of a value 1, 2, 4, 8, 16, 1/16, 1/4 or 1/2.

24. The method of anyone of claims 21-23, wherein *M1 and M2* is signaled in at least one of a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.

25. The method of anyone of claims 21-24, wherein *M1 and M2* is signaled based on the third indication.

26. The method of anyone of claims 21-24, wherein *M1 and M2* is based on a coded mode which is one of a non-affine inter mode, an affine inter mode and an IBC mode.

27 The method of anyone of claims 1-26, further comprising:

determining, based on a fifth indication and a sixth indication in the bitstream representation of the video, a non-default motion precision for motion information used for the current video block, wherein adaptive motion vector resolution (AMVR) is enabled for the current video block; and

performing, the conversion between the current video block and the bitstream representation of the current video block, based on the determined motion precision.

28. The method of claim 27, wherein the fifth indication is same as the first indication.

29. The method of claim 27 or 28, wherein the sixth indication is an *amvr\_precision\_flag*.

30. The method of claim 29, wherein the *amvr\_precision\_flag* with a value of 0 indicates the motion precision is *L0* luma samples for a non-affine inter mode and *L1* luma samples for an affine inter mode, in response to the fifth indication indicates that subpel motion precision is disallowed.

31. The method of claim 29, wherein the *amvr\_precision\_flag* with a value of 1 indicates the motion precision is *L2* luma samples for a non-affine inter mode and *L3* luma samples for an affine inter mode, in response to the fifth indication indicates that subpel motion precision is allowed.

32. The method of claim 29, wherein the *amvr\_precision\_flag* with a value of 0 indicates the motion precision is *L0* luma samples for a non-affine inter mode and *L1* luma samples for an affine inter mode, in response to the fifth indication indicates that a current video block is not a screen content.

33. The method of claim 29, wherein the `amvr_precision_flag` with a value of 1 indicates the motion precision is  $L2$  luma samples for a non-affine inter mode and  $L3$  luma samples for an affine inter mode, in response to the fifth indication indicates that a current video block is a screen content.

34. The method of claim 33, wherein anyone of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  is of a value from 1, 2, 4, 8, 16,  $1/16$ ,  $1/4$ ,  $1/2$ .

35. The method of claim 33, wherein anyone of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  is signaled in at least one of a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), a tile group header, a tile header, or a slice header.

36. The method of claim 33, wherein anyone of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  is signaled based on the fifth indication.

37. The method of claim 33, wherein anyone of  $L0$ ,  $L1$ ,  $L2$ ,  $L3$  is based on a coded mode which is one of a non-affine inter mode, an affine inter mode and an IBC mode.

38. The method of anyone of claims 1-37, wherein a first set of allowed motion precisions for a non-affine inter mode of the screen content is same with a second set of allowed motion precisions for a non-affine inter mode of a none screen content.

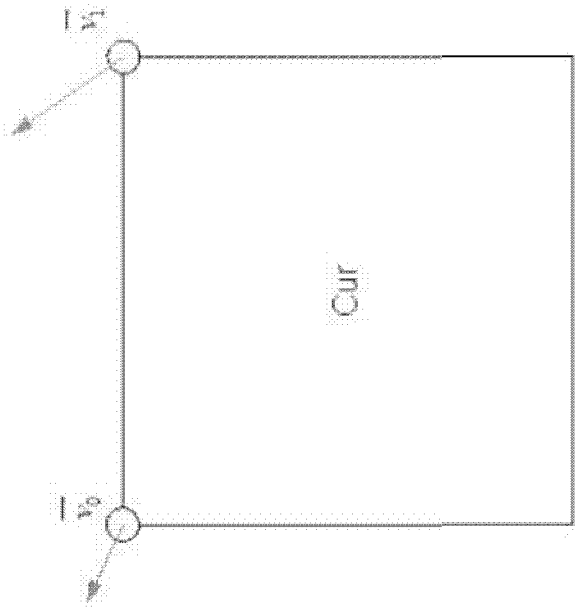
39. The method of anyone of claims 1-37, wherein a first set of allowed motion precisions for an affine inter mode of the screen content is same with a second set of allowed motion precisions for an affine inter mode of a none screen content.

40. A video processing apparatus comprising a processor configured to implement a method recited in anyone of claims 1 to 39.

41. The apparatus of claim 40, wherein the apparatus is a video encoder.

42. The apparatus of claim 40, wherein the apparatus is a video decoder.

43. A computer readable recoding media on which a program comprising code is recorded, the program is for a processor to carry out a method recited in anyone of claims 1 to 39.



Simplified affine motion model

FIG. 1

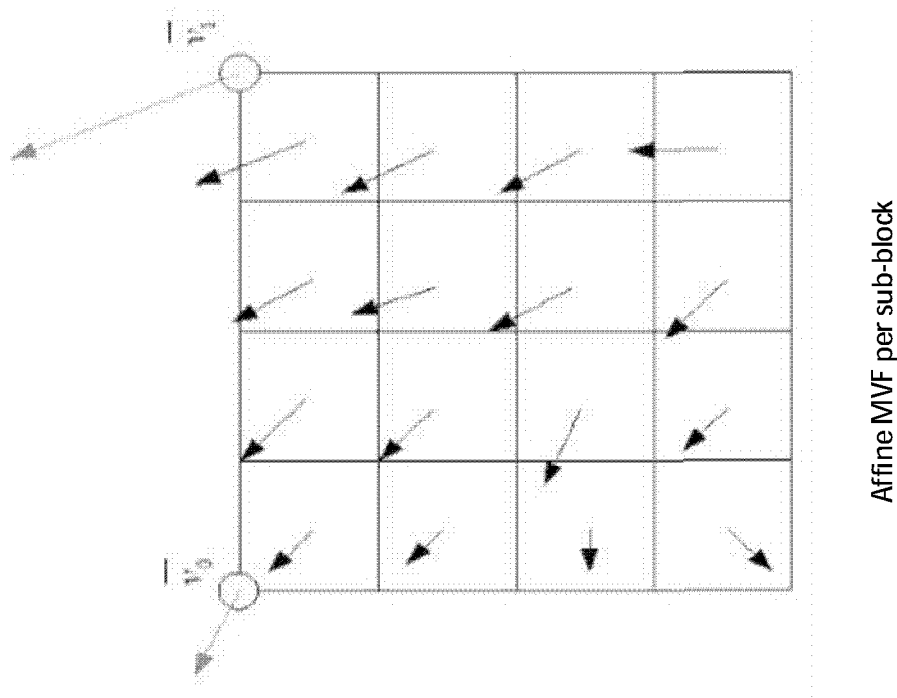
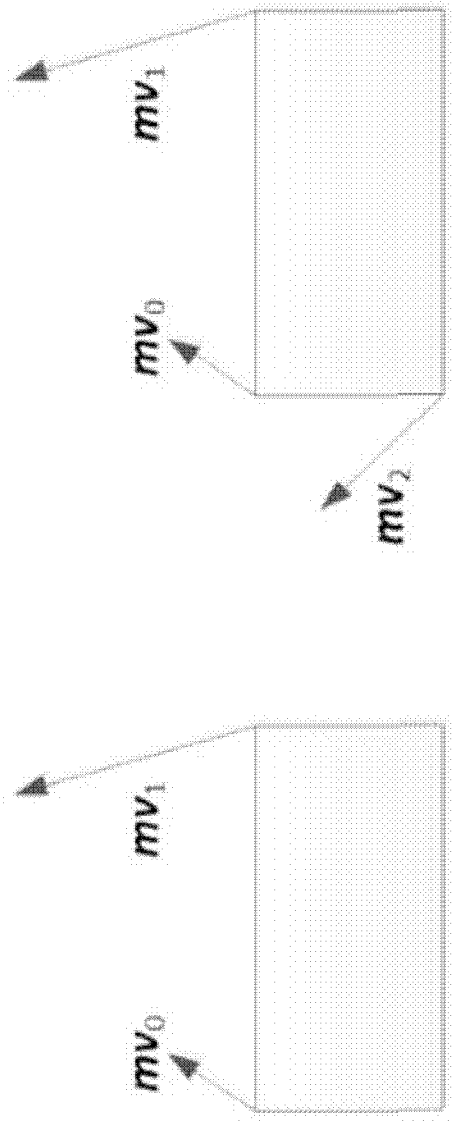


FIG. 2

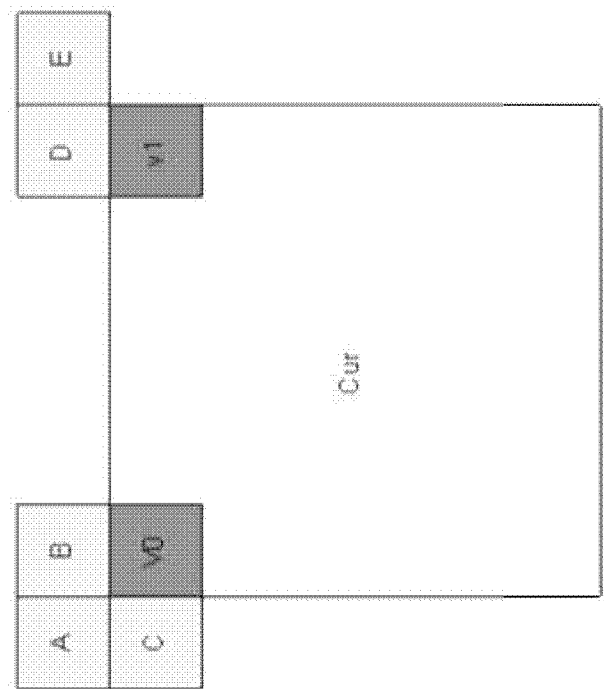


6-parameter affine model

FIG. 3B

4-parameter affine model

FIG. 3A



MVP for AF\_INTER

FIG. 4

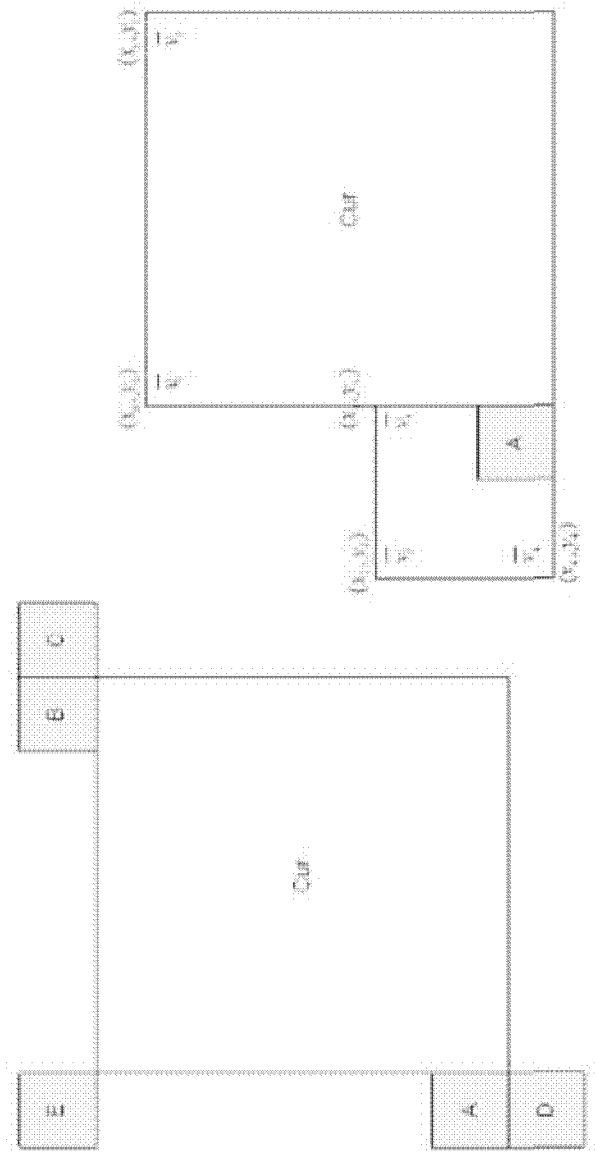
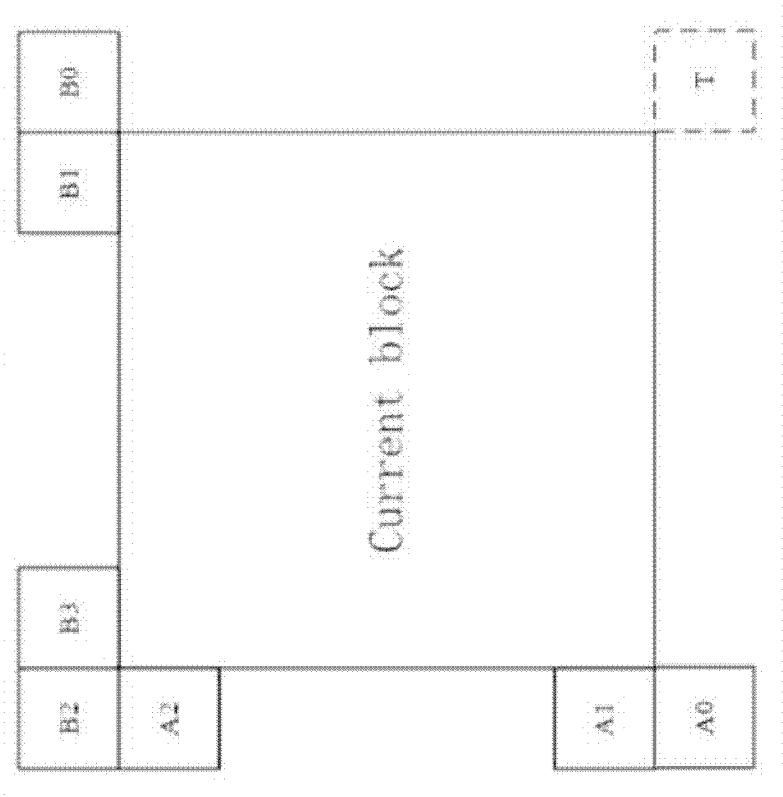


FIG. 5B

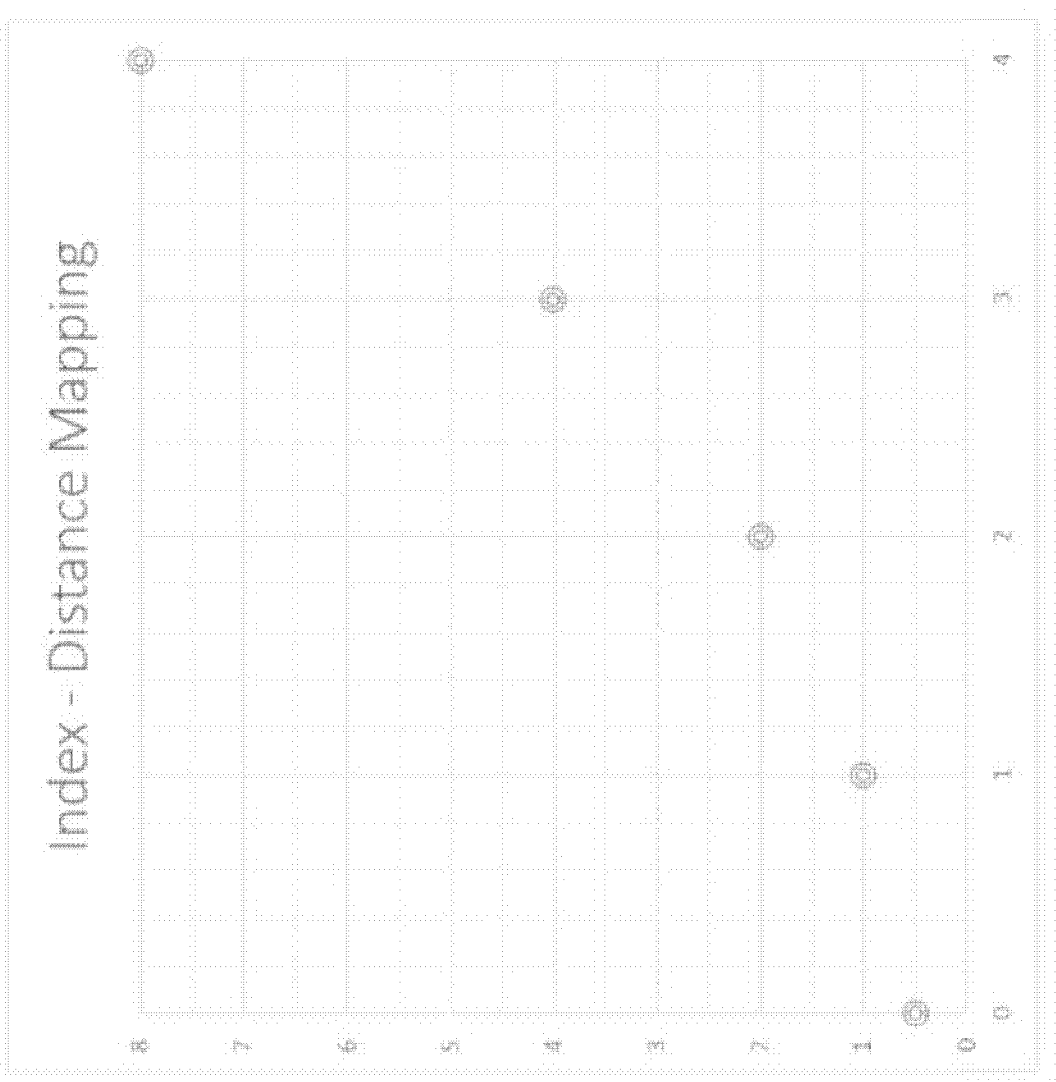
Candidates for AF\_MERGE

FIG. 5A



Candidates position for affine merge mode

FIG. 6



Distance index and distance offset mapping

FIG. 7

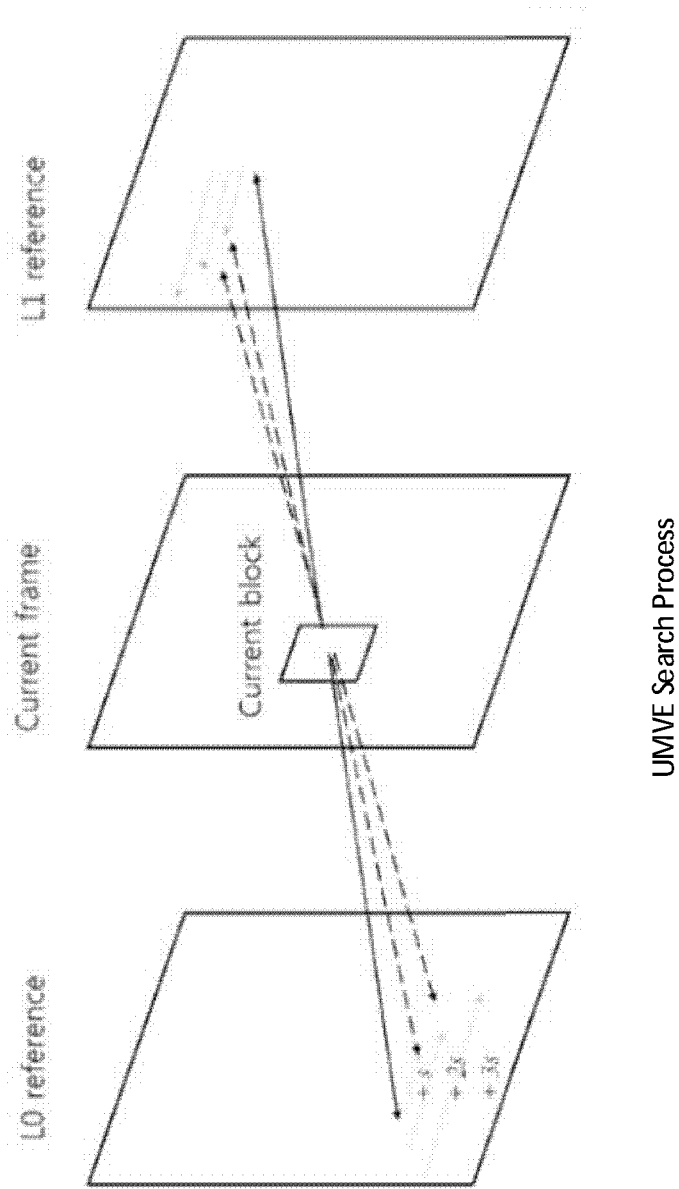


FIG. 8

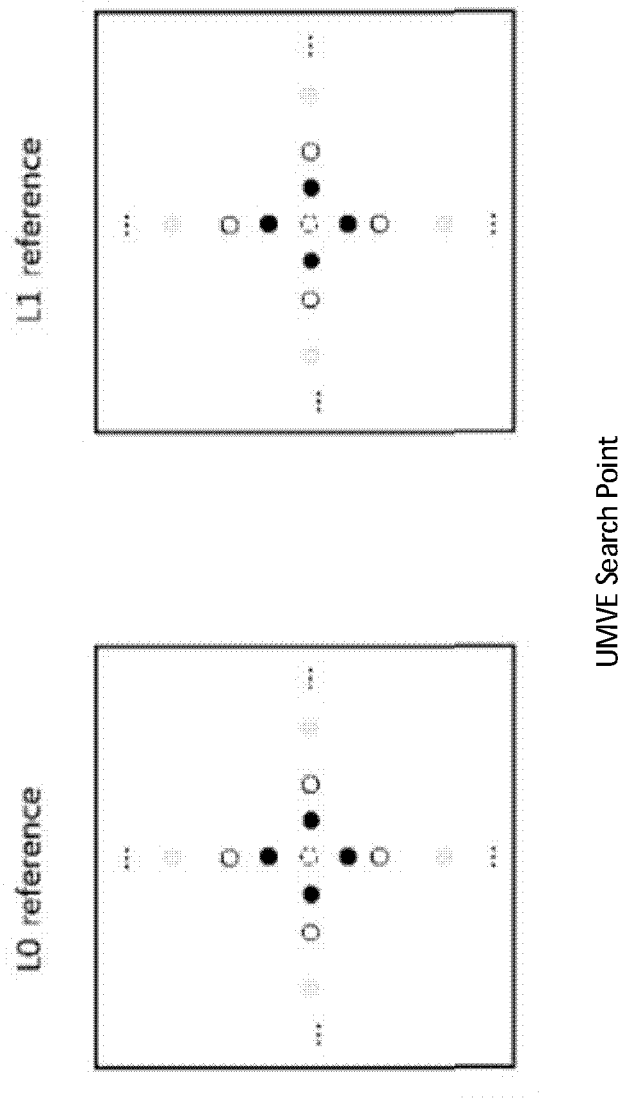


FIG. 9

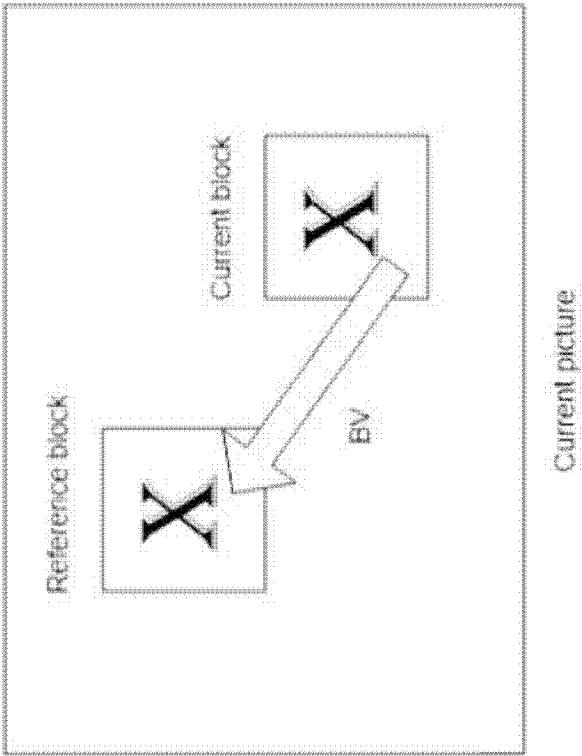
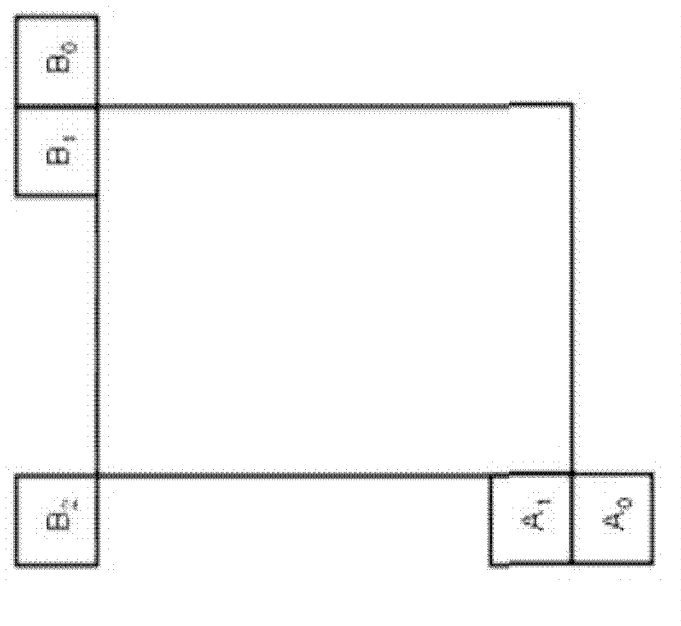


Illustration of Intra block copy

FIG. 10



Positions of spatial merge candidates

FIG. 11

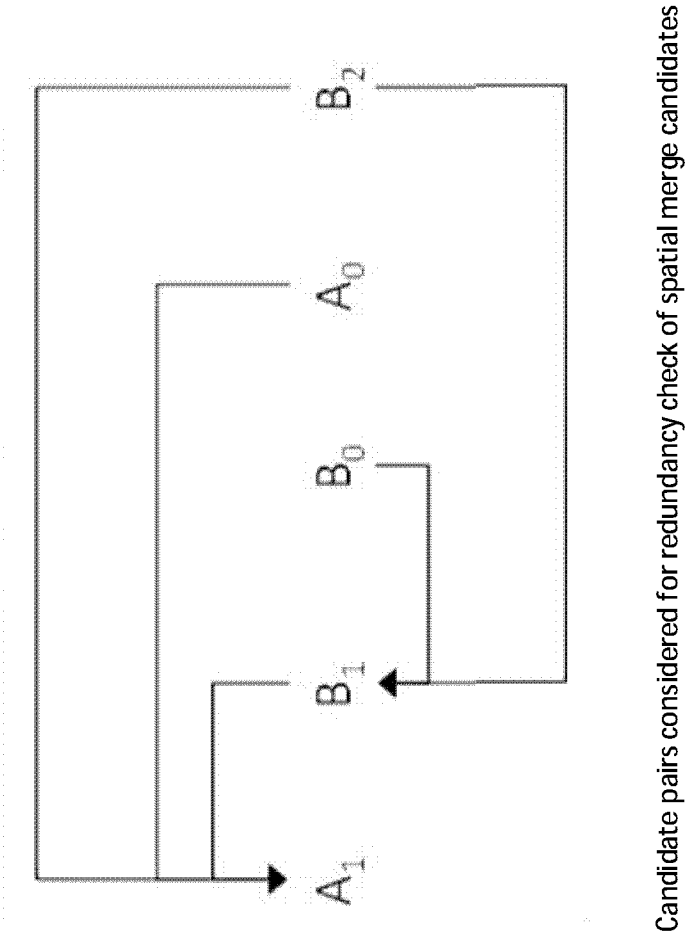
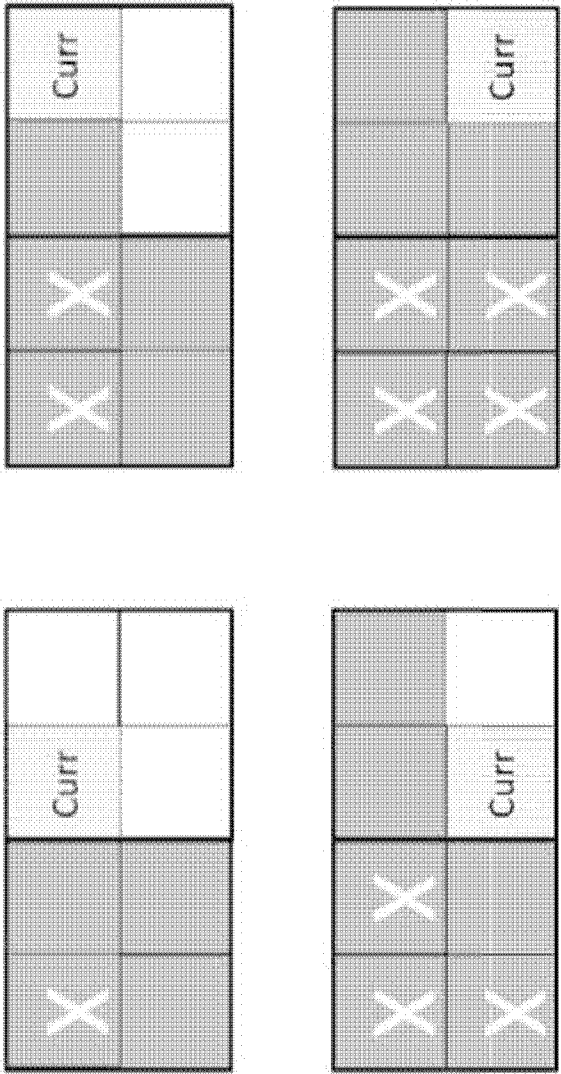


FIG. 12



current CTU processing order and its available reference samples in current and left CTU  
(when quad-tree split is enforced at 128x128 level)

FIG. 13

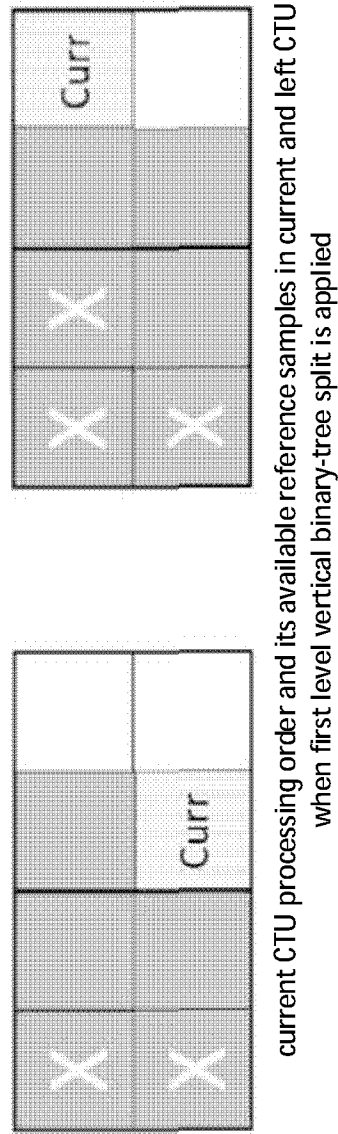


FIG. 14

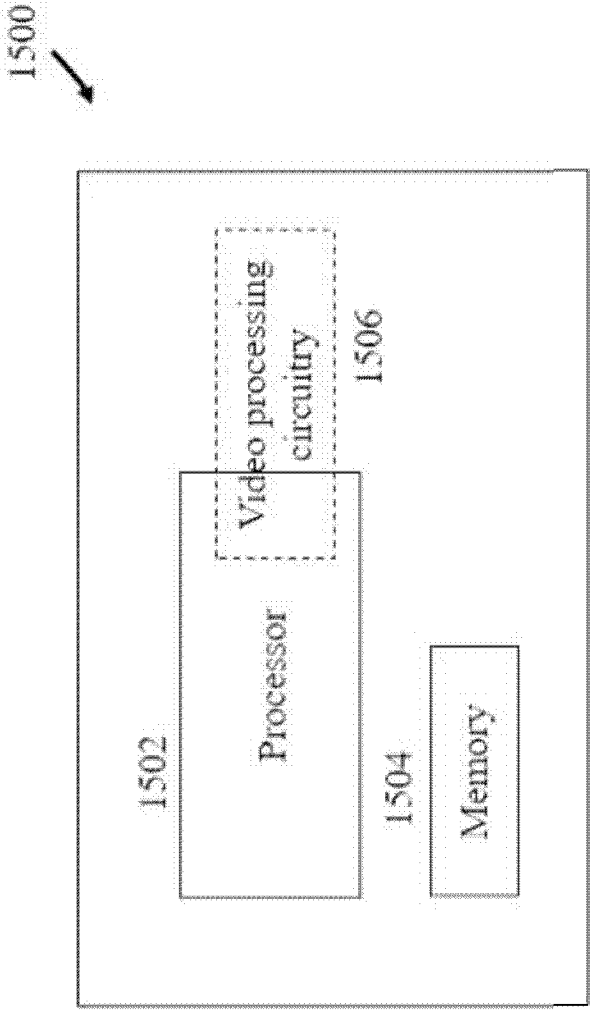


FIG. 15

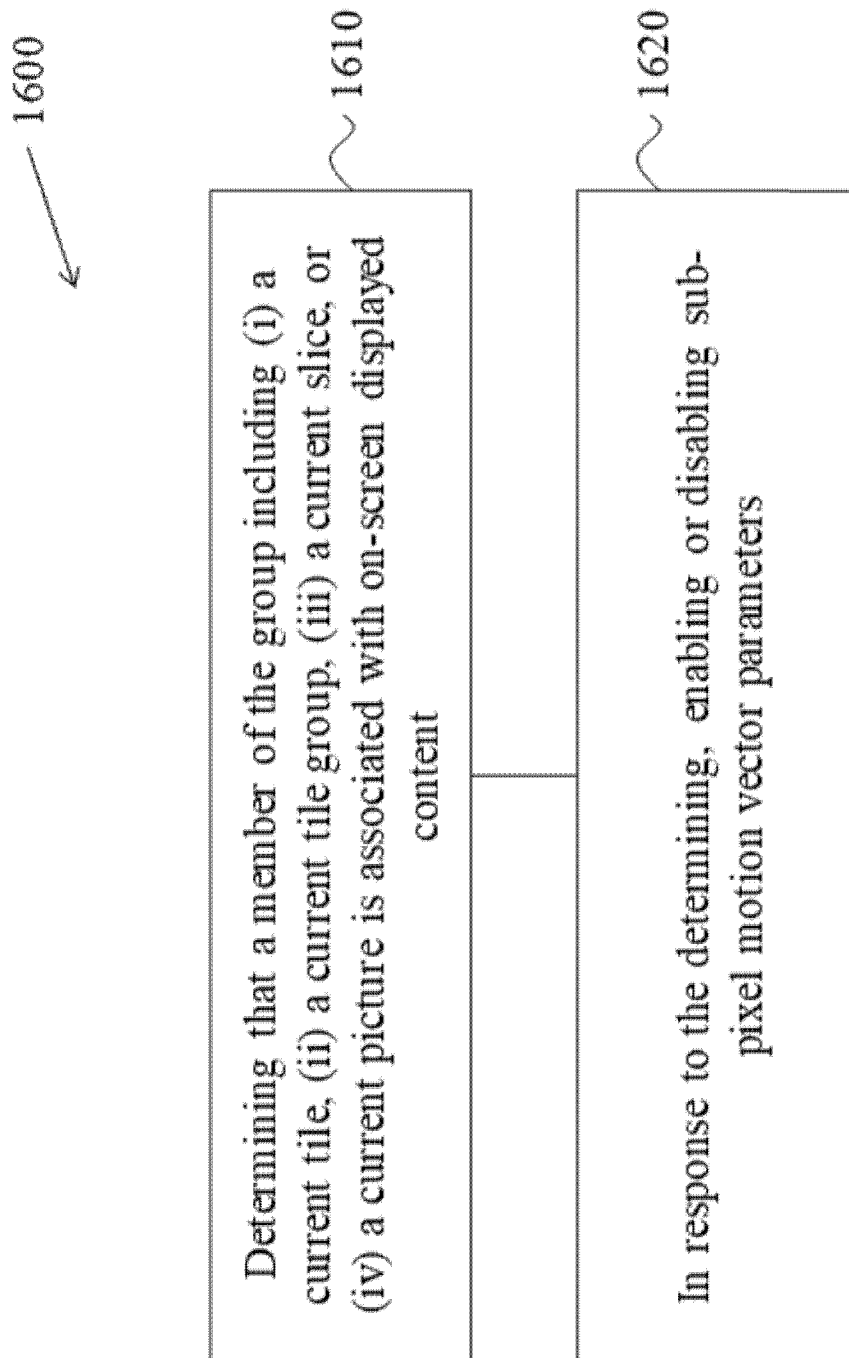


FIG. 16

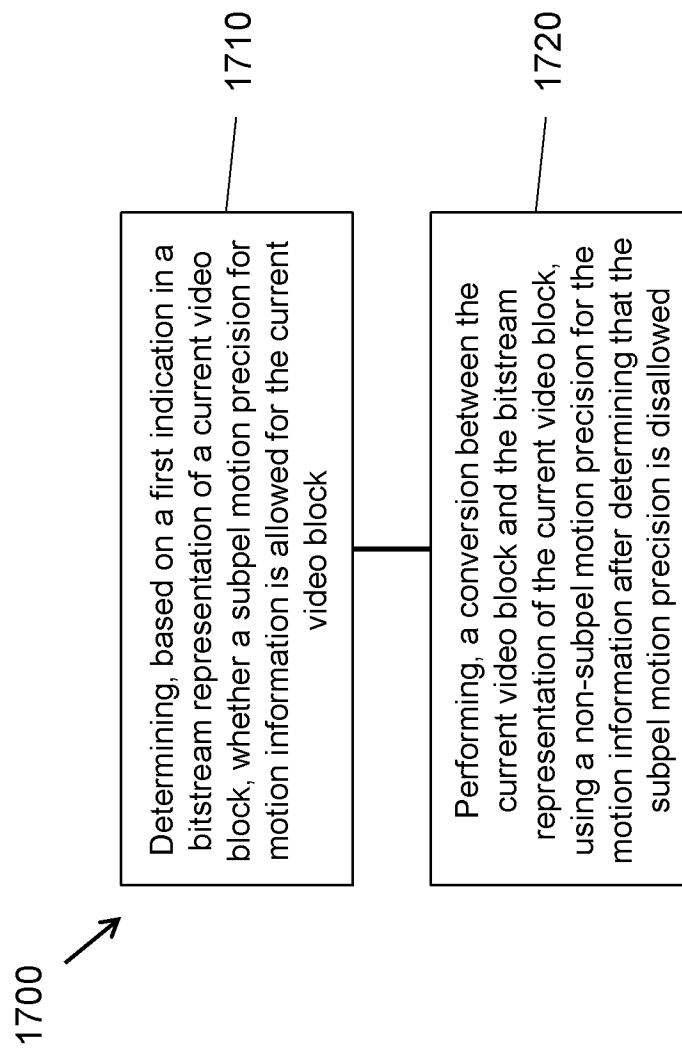


FIG. 17

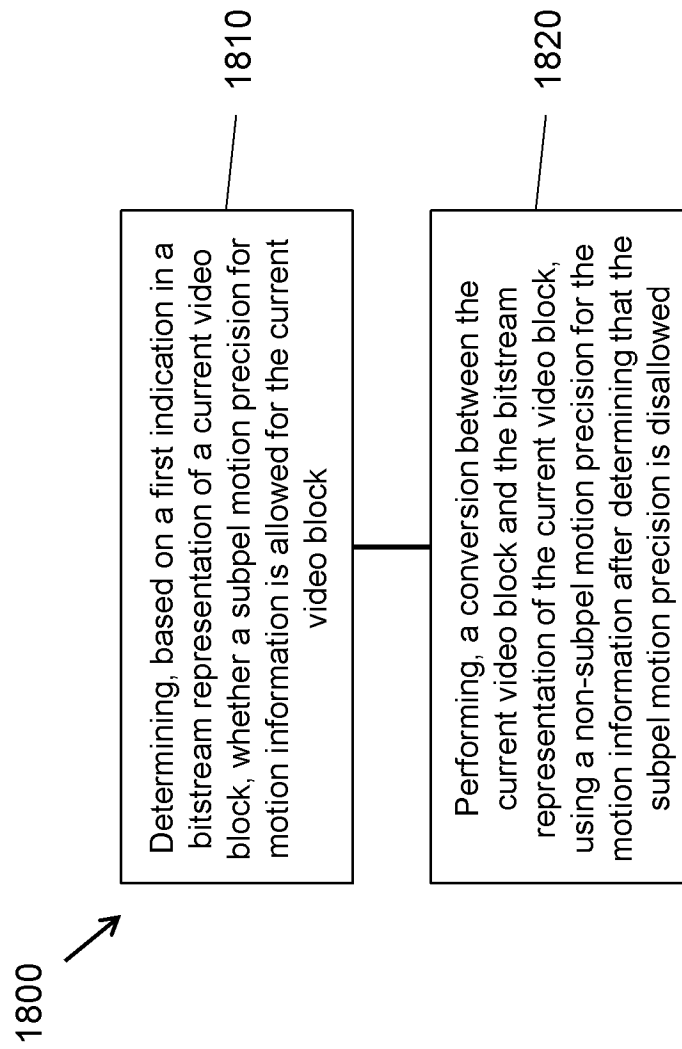


FIG. 18

## INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/076981

**A. CLASSIFICATION OF SUBJECT MATTER**

H04N 19/52(2014.01)i

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

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)

H04N

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

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

CNABS, CNKI, CNTXT, VEN, WOTXT, USTXT, EPTXT: subpel, sub, pixel, disallow, enable, disable, precision, motion, vector, merge, mode, difference, MMVD, adaptive, AMVR, flag, indicator, indication, luma, brightness, luminance, sample

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

| Category* | Citation of document, with indication, where appropriate, of the relevant passages              | Relevant to claim No. |
|-----------|-------------------------------------------------------------------------------------------------|-----------------------|
| X         | CN 104704835 A (MEDIATEK INC) 10 June 2015 (2015-06-10)<br>description paragraphs [0002]-[0039] | 1-43                  |
| A         | US 2014023142 A1 (GEN INSTRUMENT CORP) 23 January 2014 (2014-01-23)<br>the whole document       | 1-43                  |
| A         | CN 107071464 A (HFI INNOVATION INC) 18 August 2017 (2017-08-18)<br>the whole document           | 1-43                  |
| A         | US 2018199055 A1 (QUALCOMM INC) 12 July 2018 (2018-07-12)<br>the whole document                 | 1-43                  |



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

07 May 2020

Date of mailing of the international search report

14 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

QI, Jingwei

Facsimile No. (86-10)62019451

Telephone No. 86- (010) -62411482

**INTERNATIONAL SEARCH REPORT**  
**Information on patent family members**

International application No.

**PCT/CN2020/076981**

| Patent document<br>cited in search report |            |    | Publication date<br>(day/month/year) | Patent family member(s) |            |    | Publication date<br>(day/month/year) |
|-------------------------------------------|------------|----|--------------------------------------|-------------------------|------------|----|--------------------------------------|
| CN                                        | 104704835  | A  | 10 June 2015                         | US                      | 2015281733 | A1 | 01 October 2015                      |
|                                           |            |    |                                      | WO                      | 2014053085 | A1 | 10 April 2014                        |
|                                           |            |    |                                      | US                      | 10178410   | B2 | 08 January 2019                      |
|                                           |            |    |                                      | CN                      | 104704835  | B  | 24 November 2017                     |
| US                                        | 2014023142 | A1 | 23 January 2014                      | US                      | 9319681    | B2 | 19 April 2016                        |
| CN                                        | 107071464  | A  | 18 August 2017                       | SG                      | 189843     | A1 | 28 June 2013                         |
|                                           |            |    |                                      | IL                      | 226236     | D0 | 31 July 2013                         |
|                                           |            |    |                                      | CN                      | 103299630  | A  | 11 September 2013                    |
|                                           |            |    |                                      | AU                      | 2012208842 | A1 | 02 May 2013                          |
|                                           |            |    |                                      | CN                      | 103299630  | B  | 19 October 2016                      |
|                                           |            |    |                                      | WO                      | 2012097749 | A1 | 26 July 2012                         |
|                                           |            |    |                                      | US                      | 2013208804 | A1 | 15 August 2013                       |
|                                           |            |    |                                      | EP                      | 2666297    | A4 | 27 January 2016                      |
|                                           |            |    |                                      | US                      | 9300963    | B2 | 29 March 2016                        |
|                                           |            |    |                                      | EP                      | 2666297    | A1 | 27 November 2013                     |
|                                           |            |    |                                      | AU                      | 2012208842 | B2 | 07 August 2014                       |
|                                           |            |    |                                      | US                      | 2016173872 | A1 | 16 June 2016                         |
|                                           |            |    |                                      | US                      | 9525879    | B2 | 20 December 2016                     |
|                                           |            |    |                                      | EP                      | 3566453    | A1 | 13 November 2019                     |
|                                           |            |    |                                      | US                      | 10542280   | B2 | 21 January 2020                      |
|                                           |            |    |                                      | CN                      | 110089116  | A  | 02 August 2019                       |
|                                           |            |    |                                      | WO                      | 2018129539 | A1 | 12 July 2018                         |
| US                                        | 2018199055 | A1 | 12 July 2018                         | EP                      | 3566453    | A1 | 13 November 2019                     |
|                                           |            |    |                                      | US                      | 10542280   | B2 | 21 January 2020                      |
|                                           |            |    |                                      | CN                      | 110089116  | A  | 02 August 2019                       |
|                                           |            |    |                                      | WO                      | 2018129539 | A1 | 12 July 2018                         |