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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 July 2024 (18.07.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/149293 A1

(51) International Patent Classification:
H04N 19/159 (2014.01) *H04N 19/176* (2014.01) *H04N 19/186* (2014.01) *H04N 19/70* (2014.01)

(21) International Application Number: PCT/CN2024/071591

(22) International Filing Date: 10 January 2024 (10.01.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/479,556 12 January 2023 (12.01.2023) US
63/487,633 01 March 2023 (01.03.2023) US

(71) Applicant: **MEDIATEK INC.** [CN/CN]; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN).

(72) Inventors: **CHUANG, Cheng-Yen**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN). **TSAI, Chia-Ming**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN). **TSENG, Hsin-Yi**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN). **CHIANG, Man-Shu**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN). **HSU, Chih-Wei**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN). **CHEN, Yi-Wen**; 2840 Junction Ave, San Jose, California 95134 (US). **CHEN, Ching-Yeh**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN). **CHUANG, Tzu-Der**; No. 1, Dusing 1st Rd., Hsinchu Science Park, Hsinchu City, Taiwan 30078 (CN).

(74) Agent: **BEIJING SANYOU INTELLECTUAL PROPERTY AGENCY LTD.**; 16th Fl., Block A, Corporate Square, No. 35 Jinrong Street, Xicheng District, Beijing 100033 (CN).

(54) **Title:** METHODS AND APPARATUS FOR IMPROVEMENT OF TRANSFORM INFORMATION CODING ACCORDING TO INTRA CHROMA CROSS-COMPONENT PREDICTION MODEL IN VIDEO CODING

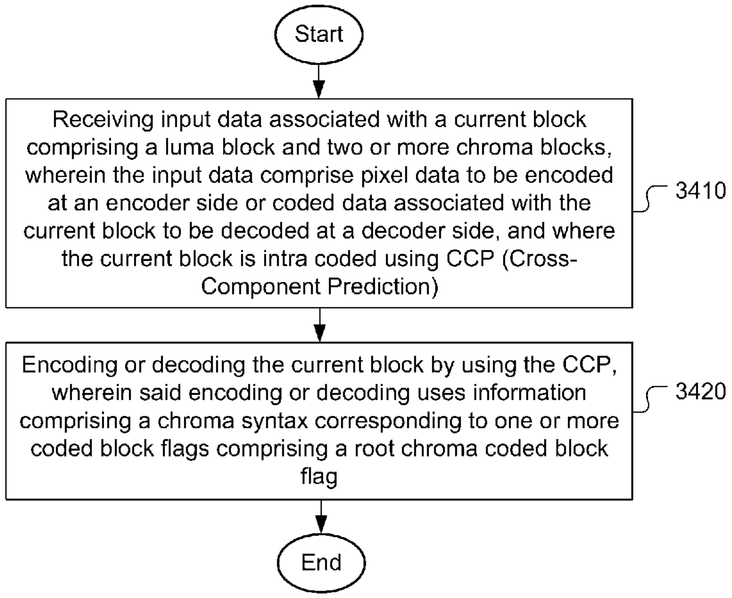

Fig. 34

(57) **Abstract:** A method and apparatus for video coding using coding tools including one or more cross component models related modes. According to the method for the decoder side, coded syntax data corresponding to one or more syntaxes related to transform coding or residual coding applied to the current block are parsed. The coded syntax data are decoded to generate said one or more syntaxes by using one or more context models according to information associated with the CCP. The current block is decoded using the CCP, wherein said decoding comprises the transform coding of the current block or the residual coding of the current block using said one or more syntaxes. At the encoder side, one or more syntaxes related to the transform coding or the residual coding are encoded to generate coded syntax data by using one or more context models according to information associated with the CCP.

WO 2024/149293 A1

[Continued on next page]

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

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

Published:

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

METHODS AND APPARATUS FOR IMPROVEMENT OF TRANSFORM INFORMATION CODING ACCORDING TO INTRA CHROMA CROSS-COMPONENT PREDICTION MODEL IN VIDEO CODING

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] The present invention is a non-Provisional Application of and claims priority to U.S. Provisional Patent Application No. 63/479,556, filed on January 12, 2023 and U.S. Provisional Patent Application No. 63/487,633, filed on March 1, 2023. The U.S. Provisional Patent Applications are hereby incorporated by reference in their entireties.

FIELD OF THE INVENTION

[0002] The present invention relates to video coding system. In particular, the present invention relates to coding syntaxes associated with transform or residual coding using one or more context models according to information associated with the cross-component prediction in a video coding system.

BACKGROUND

[0003] Versatile video coding (VVC) is the latest international video coding standard developed by the Joint Video Experts Team (JVET) of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). The standard has been published as an ISO standard: ISO/IEC 23090-3:2021, Information technology - Coded representation of immersive media - Part 3: Versatile video coding, published Feb. 2021. VVC is developed based on its predecessor HEVC (High Efficiency Video Coding) by adding more coding tools to improve coding efficiency and also to handle various types of video sources including 3-dimensional (3D) video signals.

[0004] Fig. 1A illustrates an exemplary adaptive Inter/Intra video encoding system incorporating loop processing. For Intra Prediction 110, the prediction data is derived based on previously coded video data in the current picture. For Inter Prediction 112, Motion Estimation (ME) is performed at the encoder side and Motion Compensation (MC) is performed based on the result of ME to provide prediction data derived from other picture(s) and motion data. Switch 114 selects Intra Prediction 110 or Inter-Prediction 112 and the selected prediction data is supplied to Adder 116 to form prediction errors, also called residues. The prediction error is then processed by Transform (T) 118 followed by Quantization (Q) 120. The transformed and quantized residues are then coded by Entropy Encoder 122 to be included in a video bitstream corresponding to the compressed video data. The bitstream associated with the transform coefficients is then packed with side information such as motion and coding modes associated with Intra prediction and Inter prediction, and other information such as parameters associated with loop filters applied to underlying image area. The side information associated with Intra Prediction 110, Inter prediction 112 and in-loop filter 130, are provided to Entropy Encoder 122 as shown in Fig. 1A. When an Inter-prediction mode is used, a reference picture or pictures have to be reconstructed at the encoder end as well. Consequently, the transformed and quantized residues are processed by Inverse Quantization (IQ) 124 and Inverse Transformation (IT) 126 to recover the residues. The residues are then added back to prediction data 136 at Reconstruction (REC) 128 to reconstruct video data. The reconstructed video data may be stored in Reference Picture Buffer 134 and used for prediction of other frames.

[0005] As shown in Fig. 1A, incoming video data undergoes a series of processing in the encoding system. The reconstructed video data from REC 128 may be subject to various impairments due to a series of processing. Accordingly, in-loop filter 130 is often applied to the reconstructed video data before the reconstructed video data are stored in the Reference Picture Buffer 134 in order to improve video quality. For example, deblocking filter (DF), Sample Adaptive Offset (SAO) and Adaptive Loop Filter (ALF) may be used. The loop filter information may need to be incorporated in the bitstream so that a decoder can properly recover the required information. Therefore, loop filter information is also provided to Entropy Encoder 122 for incorporation into the bitstream. In Fig. 1A, Loop filter 130 is applied to the reconstructed video before the reconstructed samples are stored in the reference picture buffer 134. The system in Fig. 1A is intended to illustrate an exemplary structure of a typical video encoder. It may correspond to the High Efficiency Video Coding (HEVC) system, VP8, VP9, H.264 or VVC.

[0006] The decoder, as shown in Fig. 1B, can use similar or portion of the same functional blocks as the encoder except for Transform 118 and Quantization 120 since the decoder only needs Inverse Quantization 124 and Inverse Transform 126. Instead of Entropy Encoder 122, the decoder uses an Entropy Decoder 140 to decode the video bitstream into quantized transform coefficients and needed coding information (e.g. ILPF information, Intra prediction information and Inter prediction information). The Intra prediction 150 at the decoder side does not need to perform the mode search. Instead, the decoder only needs to generate Intra prediction according to Intra prediction information received from the Entropy Decoder 140. Furthermore, for Inter prediction, the decoder only needs to perform motion compensation (MC 152) according to Inter prediction information received from the Entropy Decoder 140 without the need for motion estimation.

[0007] According to VVC, an input picture is partitioned into non-overlapped square block regions referred as CTUs (Coding Tree Units), similar to HEVC. Each CTU can be partitioned into one or multiple smaller size coding units (CUs). The resulting CU partitions can be in square or rectangular shapes. Also, VVC divides a CTU into prediction units (PUs) as a unit to apply prediction process, such as Inter prediction, Intra prediction, etc.

[0008] The VVC standard incorporates various new coding tools to further improve the coding efficiency over the HEVC standard. Some new tools relevant to the present invention are reviewed as follows.

[0009] **Partitioning of the CTUs Using a Tree Structure**

[0010] In HEVC, a CTU is split into CUs by using a quaternary-tree (QT) structure denoted as coding tree to adapt to various local characteristics. The decision whether to code a picture area using inter-picture (temporal) or intra-picture (spatial) prediction is made at the leaf CU level. Each leaf CU can be further split into one, two or four Pus according to the PU splitting type. Inside one PU, the same prediction process is applied and the relevant information is transmitted to the decoder on a PU basis. After obtaining the residual block by applying the prediction process based on the PU splitting type, a leaf CU can be partitioned into transform units (TUs) according to another quaternary-tree structure similar to the coding tree for the CU. One of key feature of the HEVC structure is that it has the multiple partition conceptions including CU, PU, and TU.

[0011] In VVC, a quadtree with nested multi-type tree using binary and ternary splits segmentation structure replaces the concepts of multiple partition unit types, i.e. it removes the separation of the CU, PU and TU concepts except as needed for CUs that have a size too large for the maximum transform length, and supports more flexibility for CU partition shapes. In the coding tree structure, a CU can have either a square or rectangular shape. A coding tree unit (CTU) is first partitioned by a quaternary tree (a.k.a. quadtree) structure. Then the quaternary tree leaf nodes can be further partitioned by a multi-type tree structure. As shown in Fig. 2, there are four splitting types in multi-type tree structure, vertical binary splitting (SPLIT_BT_VER 210), horizontal binary splitting (SPLIT_BT_HOR 220), vertical ternary splitting (SPLIT_TT_VER 230), and horizontal ternary splitting (SPLIT_TT_HOR 240). The multi-type tree leaf nodes are called coding units (CUs), and unless the CU is too large for the maximum transform length, this segmentation is used for prediction and transform processing without any further partitioning. This means that, in most cases, the CU, PU and TU have the same block size in the quadtree with nested multi-type tree coding block structure. The exception occurs when maximum supported transform length is smaller than the width or height of the colour component of the CU.

[0012] Fig. 3 illustrates the signalling mechanism of the partition splitting information in quadtree with nested multi-type tree coding tree structure. A coding tree unit (CTU) is treated as the root of a quaternary tree and is first partitioned by a quaternary tree structure. Each quaternary tree leaf node (when sufficiently large to allow it) is then further partitioned by a multi-type tree structure. In quadtree with nested multi-type tree coding tree structure, for each CU node, a first flag (split_cu_flag) is signalled to indicate whether the node is further partitioned. If the current CU node is a quadtree CU node, a second flag (split_qt_flag) whether it's a QT partitioning or MTT partitioning mode. When a node is partitioned with MTT partitioning mode, a third flag (mtt_split_cu_vertical_flag) is signalled to indicate the splitting direction, and then a fourth flag (mtt_split_cu_binary_flag) is signalled to indicate whether the split is a binary split or a ternary split. Based on the values of mtt_split_cu_vertical_flag and mtt_split_cu_binary_flag, the multi-type tree slitting mode (MttSplitMode) of a CU is derived as shown in Table 1.

Table 1 – MttSplitMode derivations based on multi-type tree syntax elements

MttSplitMode	mtt_split_cu_vertical_flag	mtt_split_cu_binary_flag
SPLIT_TT_HOR	0	0
SPLIT_BT_HOR	0	1
SPLIT_TT_VER	1	0
SPLIT_BT_VER	1	1

[0013] Fig. 4 shows a CTU divided into multiple CUs with a quadtree and nested multi-type tree coding block structure, where the bold block edges represent quadtree partitioning and the remaining edges represent multi-type tree partitioning. The quadtree with nested multi-type tree partition provides a content-adaptive coding tree structure comprised of CUs. The size of the CU may be as large as the CTU or as small as 4×4 in units of luma samples. For the case of the 4:2:0 chroma format, the maximum chroma CB size is 64×64 and the minimum size chroma CB consist of 16 chroma samples.

[0014] In VVC, the maximum supported luma transform size is 64×64 and the maximum supported chroma transform size is 32×32. When the width or height of the CB is larger the maximum transform width or height, the CB is automatically split in the horizontal and/or vertical direction to meet the transform size restriction in that direction.

[0015] The following parameters are defined for the quadtree with nested multi-type tree coding tree scheme. These parameters are specified by SPS (Sequence Parameter Set) syntax elements and can be further refined by picture header syntax elements.

- CTU size: the root node size of a quaternary tree
- *MinQTSize*: the minimum allowed quaternary tree leaf node size
- *MaxBtSize*: the maximum allowed binary tree root node size
- *MaxTtSize*: the maximum allowed ternary tree root node size
- *MaxMttDepth*: the maximum allowed hierarchy depth of multi-type tree splitting from a quadtree leaf
- *MinCbSize*: the minimum allowed coding block node size

[0016] In one example of the quadtree with nested multi-type tree coding tree structure, the CTU size is set as 128×128 luma samples with two corresponding 64×64 blocks of 4:2:0 chroma samples, the *MinQTSize* is set as 16×16, the *MaxBtSize* is set as 128×128 and *MaxTtSize* is set as 64×64, the *MinCbsize* (for both width and height) is set as 4×4, and the *MaxMttDepth* is set as 4. The quaternary tree partitioning is applied to the CTU first to generate quaternary tree leaf nodes. The quaternary tree leaf nodes may have a size from 16×16 (i.e., the *MinQTSize*) to 128×128 (i.e., the

CTU size). If the leaf QT node is 128×128, it will not be further split by the binary tree since the size exceeds the *MaxBtSize* and *MaxTtSize* (i.e., 64×64). Otherwise, the leaf qdtree node can be further partitioned by the multi-type tree. Therefore, the quaternary tree leaf node is also the root node for the multi-type tree and it has multi-type tree depth (mttDepth) as 0. When the multi-type tree depth reaches *MaxMttDepth* (i.e., 4), no further splitting is considered. When the multi-type tree node has width equal to *MinCbsize*, no further horizontal splitting is considered. Similarly, when the multi-type tree node has height equal to *MinCbsize*, no further vertical splitting is considered.

[0017] In VVC, the coding tree scheme supports the ability for the luma and chroma to have a separate block tree structure. For P and B slices, the luma and chroma CTBs in one CTU have to share the same coding tree structure. However, for I slices, the luma and chroma can have separate block tree structures. When the separate block tree mode is applied, luma CTB is partitioned into CUs by one coding tree structure, and the chroma CTBs are partitioned into chroma CUs by another coding tree structure. This means that a CU in an I slice may consist of a coding block of the luma component or coding blocks of two chroma components, and a CU in a P or B slice always consists of coding blocks of all three colour components unless the video is monochrome.

[0018] **Virtual Pipeline Data Units (VPDUs)**

[0019] Virtual pipeline data units (VPDUs) are defined as non-overlapping units in a picture. In hardware decoders, successive VPDUs are processed by multiple pipeline stages at the same time. The VPDU size is roughly proportional to the buffer size in most pipeline stages, so it is important to keep the VPDU size small. In most hardware decoders, the VPDU size can be set to maximum transform block (TB) size. However, in VVC, ternary tree (TT) and binary tree (BT) partition may lead to the increasing of VPDUs size.

[0020] In order to keep the VPDU size as 64x64 luma samples, the following normative partition restrictions (with syntax signalling modification) are applied in VTM, as shown in Fig. 5:

- TT split is not allowed (as indicated by "X" in Fig. 5) for a CU with either width or height, or both width and height equal to 128.
- For a 128xN CU with N ≤ 64 (i.e. width equal to 128 and height smaller than 128), horizontal BT is not allowed.

[0021] For an Nx128 CU with N ≤ 64 (i.e. height equal to 128 and width smaller than 128), vertical BT is not allowed. In Fig. 5, the luma block size is 128x128. The dashed lines indicate block size 64x64. According to the constraints mentioned above, examples of the partitions not allowed are indicated by "X" as shown in various examples (510-580) in Fig. 5.

[0022] **Intra Chroma Partitioning and Prediction Restriction**

[0023] In typical hardware video encoders and decoders, processing throughput drops when a picture has smaller intra blocks because of sample processing data dependency between neighbouring intra blocks. The predictor generation of an intra block requires top and left boundary reconstructed samples from neighbouring blocks. Therefore, intra prediction has to be sequentially processed block by block.

[0024] In HEVC, the smallest intra CU is 8x8 luma samples. The luma component of the smallest intra CU can be further split into four 4x4 luma intra prediction units (PUs), but the chroma components of the smallest intra CU cannot be further split. Therefore, the worst case hardware processing throughput occurs when 4x4 chroma intra blocks or 4x4 luma intra blocks are processed. In VVC, in order to improve worst case throughput, chroma intra CBs smaller than 16 chroma samples (size 2x2, 4x2, and 2x4) and chroma intra CBs with width smaller than 4 chroma samples (size 2xN) are disallowed by constraining the partitioning of chroma intra CBs.

[0025] In single coding tree, a smallest chroma intra prediction unit (SCIPU) is defined as a coding tree node whose chroma block size is larger than or equal to 16 chroma samples and has at least one child luma block smaller than 64 luma samples, or a coding tree node whose chroma block size is not 2xN and has at least one child luma block 4xN luma samples. It is required that in each SCIPU, all CBs are inter, or all CBs are non-inter, i.e., either intra or intra block copy (IBC). In case of a non-inter SCIPU, it is further required that chroma of the non-inter SCIPU shall not be further split and luma of the SCIPU is allowed to be further split. In this way, the small chroma intra CBs with size less than 16 chroma samples or with size 2xN are removed. In addition, chroma scaling is not applied in case of a non-inter SCIPU. Here, no additional syntax is signalled, and whether a SCIPU is non-inter can be derived by the prediction mode of the first luma CB in the SCIPU. The type of a SCIPU is inferred to be non-inter if the current slice is an I-slice or the current SCIPU has a 4x4 luma partition in it after further split one time (because no inter 4x4 is allowed in VVC); otherwise, the type of the SCIPU (inter or non-inter) is indicated by one flag before parsing the CUs in the SCIPU.

[0026] For the dual tree in intra picture, the 2xN intra chroma blocks are removed by disabling vertical binary and vertical ternary splits for 4xN and 8xN chroma partitions, respectively. The small chroma blocks with sizes 2x2, 4x2, and 2x4 are also removed by partitioning restrictions.

[0027] In addition, a restriction on picture size is considered to avoid 2x2/2x4/4x2/2xN intra chroma blocks at the corner of pictures by considering the picture width and height to be multiple of max (8, MinCbSizeY).

[0028] **Intra Mode Coding with 67 Intra Prediction Modes**

[0029] To capture the arbitrary edge directions presented in natural video, the number of directional intra modes in VVC is extended from 33, as used in HEVC, to 65. The new directional modes not in HEVC are depicted as red dotted arrows in Fig. 6, and the planar and DC modes remain the same. These denser directional intra prediction modes apply for all block sizes and for both luma and chroma intra predictions.

[0030] In VVC, several conventional angular intra prediction modes are adaptively replaced with wide-angle intra prediction modes for the non-square blocks.

[0031] In HEVC, every intra-coded block has a square shape and the length of each of its side is a power of 2. Thus, no division operations are required to generate an intra-predictor using DC mode. In VVC, blocks can have a rectangular shape that necessitates the use of a division operation per block in the general case. To avoid division operations for DC prediction, only the longer side is used to compute the average for non-square blocks.

[0032] To keep the complexity of the most probable mode (MPM) list generation low, an intra mode coding method with 6 MPMs is used by considering two available neighbouring intra modes. The following three aspects are considered to construct the MPM list:

– Default intra modes
– Neighbouring intra modes
– Derived intra modes.

[0033] A unified 6-MPM list is used for intra blocks irrespective of whether MRL and ISP coding tools are applied or not. The MPM list is constructed based on intra modes of the left and above neighbouring block. Suppose the mode of the left is denoted as Left and the mode of the above block is denoted as *Above*, the unified MPM list is constructed as follows:

– When a neighbouring block is not available, its intra mode is set to Planar by default.
– If both modes Left and Above are non-angular modes:
    – MPM list → {Planar, DC, V, H, V – 4, V + 4}
– If one of modes Left and Above is angular mode, and the other is non-angular:
    – Set a mode Max as the larger mode in Left and Above
    – MPM list → {Planar, Max, Max – 1, Max + 1, Max – 2, Max + 2}
– If Left and Above are both angular and they are different:
    – Set a mode Max as the larger mode in Left and Above
  – If Max – Min is equal to 1:
    • MPM list → {Planar, Left, Above, Min – 1, Max + 1, Min – 2}
  – Otherwise, if Max – Min is greater than or equal to 62:
    • MPM list → {Planar, Left, Above, Min + 1, Max – 1, Min + 2}
  – Otherwise, if Max – Min is equal to 2:
    • MPM list → {Planar, Left, Above, Min + 1, Min – 1, Max + 1}
  – Otherwise:

- MPM list → {Planar, Left, Above, Min – 1, Min + 1, Max – 1}

– If Left and Above are both angular and they are the same:

  – MPM list → {Planar, Left, Left − 1, Left + 1, Left – 2, Left + 2}

[0034] Besides, the first bin of the MPM index codeword is CABAC context coded. In total three contexts are used, corresponding to whether the current intra block is MRL enabled, ISP enabled, or a normal intra block.

[0035] During 6 MPM list generation process, pruning is used to remove duplicated modes so that only unique modes can be included into the MPM list. For entropy coding of the 61 non-MPM modes, a Truncated Binary Code (TBC) is used.

[0036] **Wide-Angle Intra Prediction for Non-Square Blocks**

[0037] Conventional angular intra prediction directions are defined from 45 degrees to −135 degrees in clockwise direction. In VVC, several conventional angular intra prediction modes are adaptively replaced with wide-angle intra prediction modes for non-square blocks. The replaced modes are signalled using the original mode indexes, which are remapped to the indexes of wide angular modes after parsing. The total number of intra prediction modes is unchanged, i.e., 67, and the intra mode coding method is unchanged.

[0038] To support these prediction directions, the top reference with length 2W+1, and the left reference with length 2H+1, are defined as shown in Fig. 7A and Fig. 7B respectively.

[0039] The number of replaced modes in wide-angular direction mode depends on the aspect ratio of a block. The replaced intra prediction modes are illustrated in Table 2.

Table 2 – Intra prediction modes replaced by wide-angular modes

Aspect ratio	Replaced intra prediction modes
W / H == 16	Modes 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,14,15
W / H == 8	Modes 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
W / H == 4	Modes 2,3,4,5,6,7,8,9,10,11
W / H == 2	Modes 2,3,4,5,6,7,
W / H == 1	None
W / H == 1/2	Modes 61,62,63,64,65,66
W / H == 1/4	Mode 57,58,59,60,61,62,63,64,65,66
W / H == 1/8	Modes 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66
W / H == 1/16	Modes 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66

[0040] In VVC, 4:2:2 and 4:4:4 chroma formats are supported as well as 4:2:0. Chroma derived mode (DM) derivation table for 4:2:2 chroma format was initially ported from HEVC extending the number of entries from 35 to 67 to align with the extension of intra prediction modes. Since HEVC specification does not support prediction angle below −135° and above 45°, luma intra prediction modes ranging from 2 to 5 are mapped to 2. Therefore, chroma DM derivation table for 4:2:2: chroma format is updated by replacing some values of the entries of the mapping table to convert prediction angle more precisely for chroma blocks.

[0041] **Cross-Component Linear Model (CCLM) Prediction**

[0042] To reduce the cross-component redundancy, a cross-component linear model (CCLM) prediction mode is used in the VVC, for which the chroma samples are predicted based on the reconstructed luma samples of the same CU by using a linear model as follows:

$$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{rec_L}'(i,j) + \beta \quad (1)$$

where $\mathrm{pred_C}(i,j)$ represents the predicted chroma samples in a CU and $\mathrm{rec_L}'(i,j)$ represents the downsampled reconstructed luma samples of the same CU.

[0043] The CCLM parameters (α and β) are derived with at most four neighbouring chroma samples and their corresponding down-sampled luma samples. Suppose the current chroma block dimensions are W×H, then W' and H' are set as

- W' = W, H' = H when CCLM_LT mode is applied;
- W' =W + H when CCLM_T mode is applied;
- H' = H + W when CCLM_L mode is applied.

[0044] The above neighbouring positions are denoted as S[0, −1]…S[W' − 1, −1] and the left neighbouring positions are denoted as S[−1, 0]…S[−1, H' − 1]. Then the four samples are selected as

- S[W' / 4, −1], S[3 * W' / 4, −1], S[−1, H' / 4], S[−1, 3 * H' / 4] when CCLM_LT mode is applied and both above and left neighbouring samples are available;
- S[W' / 8, −1], S[3 * W' / 8, −1], S[5 * W' / 8, −1], S[7 * W' / 8, −1] when CCLM_T mode is applied or only the above neighbouring samples are available;
- S[−1, H' / 8], S[−1, 3 * H' / 8], S[−1, 5 * H' / 8], S[−1, 7 * H' / 8] when CCLM_L mode is applied or only the left neighbouring samples are available.

[0045] The four neighbouring luma samples at the selected positions are down-sampled and compared four times to find two larger values: x^0_A and x^1_A, and two smaller values: x^0_B and x^1_B. Their corresponding chroma sample values are denoted as y^0_A, y^1_A, y^0_B and y^1_B. Then x_A, x_B, y_A and y_B are derived as:

$$x_A=(x^0_A + x^1_A +1)>>1;$$

$$x_B=(x^0_B + x^1_B +1)>>1;$$

$$y_A=(y^0_A + y^1_A +1)>>1;$$

$$y_B=(y^0_B + y^1_B +1)>>1 \quad (2)$$

[0046] Finally, the linear model parameters α and β are obtained according to the following equations.

$$\alpha = \frac{y_A - y_B}{x_A - x_B} \quad (3)$$

$$\beta = y_B - \alpha \cdot x_B \quad (4)$$

[0047] Fig. 8 shows an example of the location of the left and above samples and the sample of the current block involved in the CCLM_LT mode. Fig. 8 shows the relative sample locations of $N \times N$ chroma block 810, the corresponding $2N \times 2N$ luma block 820 and their neighbouring samples (shown as filled circles).

[0048] The division operation to calculate parameter α is implemented with a look-up table. To reduce the memory required for storing the table, the *diff* value (difference between maximum and minimum values) and the parameter α are expressed by an exponential notation. For example, *diff* is approximated with a 4-bit significant part and an exponent. Consequently, the table for 1/diff is reduced into 16 elements for 16 values of the significand as follows:

DivTable [ ] = { 0, 7, 6, 5, 5, 4, 4, 3, 3, 2, 2, 1, 1, 1, 1, 0 } (5)

[0049] This would have a benefit of both reducing the complexity of the calculation as well as the memory size required for storing the needed tables.

[0050] Besides the above template and left template can be used to calculate the linear model coefficients together, they also can be used alternatively in the other 2 LM modes, called CCLM_T, and CCLM_L modes.

[0051] In CCLM_T mode, only the above template is used to calculate the linear model coefficients. To get more samples, the above template is extended to (W+H) samples. In CCLM_L mode, only left template are used to calculate the linear model coefficients. To get more samples, the left template is extended to (H+W) samples.

[0052] In CCLM__LT mode, left and above templates are used to calculate the linear model coefficients.

[0053] To match the chroma sample locations for 4:2:0 video sequences, two types of down-sampling filter are applied to luma samples to achieve 2 to 1 down-sampling ratio in both horizontal and vertical directions. The selection of down-sampling filter is specified by a SPS level flag. The two down-sampling filters are as follows, which are corresponding to “type-0” and “type-2” content, respectively.

$$\mathrm{Rec_L}'(i,j) = [\mathrm{rec_L}(2i-1,2j-1) + 2\cdot\mathrm{rec_L}(2i,2j-1) + \mathrm{rec_L}(2i+1,2j-1) + \mathrm{rec_L}(2i-1,2j) + 2\cdot\mathrm{rec_L}(2i,2j) + \mathrm{rec_L}(2i+1,2j) + 4] \gg 3 \quad (6)$$

$$\text{Rec}_L'(i,j) = \text{rec}_L(2i,2j-1) + \text{rec}_L(2i-1,2j) + 4 \cdot \text{rec}_L(2i,2j) + \text{rec}_L(2i+1,2j) + \text{rec}_L(2i,2j+1) + 4] \gg 3 \quad (7)$$

[0054] Note that only one luma line (general line buffer in intra prediction) is used to make the down-sampled luma samples when the upper reference line is at the CTU boundary.

[0055] This parameter computation is performed as part of the decoding process, and is not just as an encoder search operation. As a result, no syntax is used to convey the α and β values to the decoder.

[0056] For chroma intra mode coding, a total of 8 intra modes are allowed for chroma intra mode coding. Those modes include five traditional intra modes and three cross-component linear model modes (CCLM_LT, CCLM_T, and CCLM_L). Chroma mode signalling and derivation process are shown in Table 3. Chroma mode coding directly depends on the intra prediction mode of the corresponding luma block. Since separate block partitioning structure for luma and chroma components is enabled in I slices, one chroma block may correspond to multiple luma blocks. Therefore, for Chroma DM mode, the intra prediction mode of the corresponding luma block covering the centre position of the current chroma block is directly inherited.

Table -3 Derivation of chroma prediction mode from luma mode when CCLM is enabled

Chroma prediction mode	Corresponding luma intra prediction mode				
	0	50	18	1	X (0 <= X <= 66)
0	66	0	0	0	0
1	50	66	50	50	50
2	18	18	66	18	18
3	1	1	1	66	1
4	0	50	18	1	X
5	81	81	81	81	81
6	82	82	82	82	82
7	83	83	83	83	83

[0057] A single binarization table is used regardless of the value of sps_cclm_enabled_flag as shown in Table 4.

Table 4– Unified binarization table for chroma prediction mode

Value of intra_chroma_pred_mode	Bin string
4	00
0	0100
1	0101
2	0110
3	0111
5	10
6	110

7	111

[0058] In Table 4, the first bin indicates whether it is regular (0) or CCLM modes (1). If it is LM mode, then the next bin indicates whether it is CCLM_LT (0) or not. If it is not CCLM_LT, next 1 bin indicates whether it is CCLM_L (0) or CCLM_T (1). For this case, when sps_cclm_enabled_flag is 0, the first bin of the binarization table for the corresponding intra_chroma_pred_mode can be discarded prior to the entropy coding. Or, in other words, the first bin is inferred to be 0 and hence not coded. This single binarization table is used for both sps_cclm_enabled_flag equal to 0 and 1 cases. The first two bins in Table 4 are context coded with its own context model, and the rest bins are bypass coded.

[0059] In addition, in order to reduce luma-chroma latency in dual tree, when the 64x64 luma coding tree node is partitioned with Not Split (and ISP is not used for the 64x64 CU) or QT, the chroma CUs in 32x32 / 32x16 chroma coding tree node are allowed to use CCLM in the following way:

– If the 32x32 chroma node is not split or partitioned QT split, all chroma CUs in the 32x32 node can use CCLM

– If the 32x32 chroma node is partitioned with Horizontal BT, and the 32x16 child node does not split or uses Vertical BT split, all chroma CUs in the 32x16 chroma node can use CCLM.

[0060] In all the other luma and chroma coding tree split conditions, CCLM is not allowed for chroma CU.

[0061] **Multiple Model CCLM (MMLM)**

[0062] In the JEM (J. Chen, E. Alshina, G. J. Sullivan, J.-R. Ohm, and J. Boyce, Algorithm Description of Joint Exploration Test Model 7, document JVET-G1001, ITU-T/ISO/IEC Joint Video Exploration Team (JVET), Jul. 2017), multiple model CCLM mode (MMLM) is proposed for using two models for predicting the chroma samples from the luma samples for the whole CU. In MMLM, neighbouring luma samples and neighbouring chroma samples of the current block are classified into two groups, each group is used as a training set to derive a linear model (i.e., a particular α and β are derived for a particular group). Furthermore, the samples of the current luma block are also classified based on the same rule for the classification of neighbouring luma samples. Three MMLM model modes (MMLM_LT, MMLM_T, and MMLM_L) are allowed for choosing the neighbouring samples from left-side and above-side, above-side only, and left-side only, respectively.

[0063] Fig. 9 shows an example of classifying the neighbouring samples into two groups. *Threshold* is calculated as the average value of the neighbouring reconstructed luma samples. A neighbouring sample with $Rec'_L[x,y]$ <= *Threshold* is classified into group 1; while a neighbouring sample with $Rec'_L[x,y]$ > *Threshold* is classified into group 2.

$$\begin{cases} Pred_C[x,y] = \alpha_1 \times Rec'_L[x,y] + \beta_1 & \text{if } Rec'_L[x,y] \le Threshold \\ Pred_C[x,y] = \alpha_2 \times Rec'_L[x,y] + \beta_2 & \text{if } Rec'_L[x,y] > Threshold \end{cases} \quad (8)$$

[0064] Accordingly, the MMLM uses two models according to the sample level of the neighbouring samples.

[0065] **Slope Adjustment of CCLM**

[0066] CCLM uses a model with 2 parameters to map luma values to chroma values as shown in Fig. 10A. The slope parameter "a" and the bias parameter "b" define the mapping as follows:

$$\text{chromaVal} = a * \text{lumaVal} + b$$

[0067] An adjustment "u" to the slope parameter is signalled to update the model to the following form, as shown in Fig. 10B:

$$\text{chromaVal} = a' * \text{lumaVal} + b'$$

where

$$a' = a + u,$$

$$b' = b - u * y_r.$$

[0068] With this selection, the mapping function is tilted or rotated around the point with luminance value y_r. The average of the reference luma samples used in the model creation as y_r in order to provide a meaningful modification to the model. Fig. 10A and 10B illustrates the process.

[0069] **Implementation of Slope Adjustment of CCLM**

[0070] Slope adjustment parameter is provided as an integer between -4 and 4, inclusive, and signalled in the bitstream. The unit of the slope adjustment parameter is (1/8)-th of a chroma sample value per luma sample value (for 10-bit content).

[0071] Adjustment is available for the CCLM models that are using reference samples both above and left of the block (e.g. "LM_CHROMA_IDX" and "MMLM_CHROMA_IDX"), but not for the "single side" modes. This selection is based on coding efficiency versus complexity trade-off considerations. "LM_CHROMA_IDX" and "MMLM_CHROMA_IDX" refers to CCLM_LT and MMLM_LT in this invention. The "single side" modes refer to CCLM_L, CCLM_T, MMLM_L, and MMLM_T in this invention.

[0072] When slope adjustment is applied for a multimode CCLM model, both models can be adjusted and thus up to two slope updates are signalled for a single chroma block.

[0073] **Encoder Approach for Slope Adjustment of CCLM**

[0074] The proposed encoder approach performs an SATD (Sum of Absolute Transformed Differences) based search for the best value of the slope update for Cr and a similar SATD based search for Cb. If either one results as a non-zero slope adjustment parameter, the combined slope adjustment pair (SATD based update for Cr, SATD based update for Cb) is included in the list of RD

(Rate-Distortion) checks for the TU.

[0075] **Convolutional Cross-Component Model (CCCM) - Single Model and Multi-Model**

[0076] In CCCM, a convolutional model is applied to improve the chroma prediction performance. The convolutional model has 7-tap filter consisting of a 5-tap plus sign shape spatial component, a nonlinear term and a bias term. The input to the spatial 5-tap component of the filter consists of a centre (C) luma sample which is collocated with the chroma sample to be predicted and its above/north (N), below/south (S), left/west (W) and right/east (E) neighbours as shown in Fig. 11.

[0077] The nonlinear term (denoted as P) is represented as power of two of the centre luma sample C and scaled to the sample value range of the content:

$$P = (C*C + midVal) >> bitDepth.$$

[0078] For example, for 10-bit contents, the nonlinear term is calculated as:

$$P = (C*C + 512) >> 10$$

[0079] The bias term (denoted as B) represents a scalar offset between the input and output (similarly to the offset term in CCLM) and is set to the middle chroma value (512 for 10-bit content).

[0080] Output of the filter is calculated as a convolution between the filter coefficients c_i and the input values and clipped to the range of valid chroma samples:

$$predChromaVal = c_0C + c_1N + c_2S + c_3E + c_4W + c_5P + c_6B$$

[0081] The filter coefficients c_i are calculated by minimising MSE between predicted and reconstructed chroma samples in the reference area. Fig. 12 illustrates an example of the reference area which consists of 6 lines of chroma samples above and left of the PU. Reference area extends one PU width to the right and one PU height below the PU boundaries. Area is adjusted to include only available samples. The extensions to the area (indicated as "paddings") are needed to support the "side samples" of the plus-shaped spatial filter in Fig. 11 and are padded when in unavailable areas.

[0082] The MSE minimization is performed by calculating autocorrelation matrix for the luma input and a cross-correlation vector between the luma input and chroma output. Autocorrelation matrix is LDL decomposed and the final filter coefficients are calculated using back-substitution. The process follows roughly the calculation of the ALF filter coefficients in ECM, however LDL decomposition was chosen instead of Cholesky decomposition to avoid using square root operations. In newer ECM, MSE minimization for CCM uses Gaussian elimination based approach.

[0083] Also, similarly to CCLM, there is an option of using a single model or multi-model variant of CCCM. The multi-model variant uses two models, one model derived for samples above the average luma reference value and another model for the rest of the samples (following the spirit of the CCLM design). Multi-model CCCM mode can be selected for PUs which have at least 128 reference samples available.

[0084] **Gradient Linear Model (GLM)**

[0085] Compared with the CCLM, instead of down-sampled luma values, the GLM utilizes luma sample gradients to derive the linear model. Specifically, when the GLM is applied, the input to the CCLM process, i.e., the down-sampled luma samples L, are replaced by luma sample gradients G. The other parts of the CCLM (e.g., parameter derivation, prediction sample linear transform) are kept unchanged.

$$C = \alpha \cdot G + \beta$$

[0086] For signalling, when the CCLM mode is enabled for the current CU, two flags are signalled separately for Cb and Cr components to indicate whether GLM is enabled for each component. If the GLM is enabled for one component, one syntax element is further signalled to select one of 4 gradient filters (1310-1340 in Fig. 13) for the gradient calculation. The GLM can be combined with the existing CCLM by signalling one extra flag in bitstream. When such combination is applied, the filter coefficients that are used to derive the input luma samples of the linear model are calculated as the combination of the selected gradient filter of the GLM and the down-sampling filter of the CCLM.

[0087] **Spatial Candidate Derivation**

[0088] The derivation of spatial merge candidates in VVC is the same as that in HEVC except that the positions of first two merge candidates are swapped. A maximum of four merge candidates (B_0, A_0, B_1 and A_1) for current CU 1410 are selected among candidates located in the positions depicted in Fig. 14. The order of derivation is B_0, A_0, B_1, A_1 and B_2. Position B_2 is considered only when one or more neighbouring CU of positions B_0, A_0, B_1, A_1 are not available (e.g. belonging to another slice or tile) or is intra coded. After candidate at position A_1 is added, the addition of the remaining candidates is subject to a redundancy check which ensures that candidates with the 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. 15 are considered and a candidate is only added to the list if the corresponding candidate used for redundancy check does not have the same motion information.

[0089] **Temporal Candidates Derivation**

[0090] In this step, only one candidate is added to the list. Particularly, in the derivation of this temporal merge candidate for a current CU 1610, a scaled motion vector is derived based on the co-located CU 1620 belonging to the collocated reference picture as shown in Fig. 16. The reference picture list and the reference index to be used for the derivation of the co-located CU is explicitly signalled in the slice header. The scaled motion vector 1630 for the temporal merge candidate is obtained as illustrated by the dotted line in Fig. 16, which is scaled from the motion vector 1640 of the co-located CU using the POC (Picture Order Count) distances, tb and td, where tb is defined to be the POC difference between the reference picture of the current picture and the current picture and td is defined to be the POC difference between the reference picture of the co-located picture and the co-located picture. The reference picture index of temporal merge candidate is set equal to zero.

[0091] The position for the temporal candidate is selected between candidates C_0 and C_1, as depicted in Fig. 17. If CU at position C_0 is not available, is intra coded, or is outside of the current row of CTUs, position C_1 is used. Otherwise, position C_0 is used in the derivation of the temporal merge candidate.

[0092] **Non-Adjacent Spatial Candidate**

[0093] During the development of the VVC standard, a coding tool referred as Non-Adjacent Motion Vector Prediction (NAMVP) has been proposed in JVET-L0399 (Yu Han, et al., "CE4.4.6: Improvement on Merge/Skip mode", Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 12th Meeting: Macao, CN, 3–12 Oct. 2018, Document: JVET-L0399). According to the NAMVP technique, the non-adjacent spatial merge candidates are inserted after the TMVP (i.e., the temporal MVP) in the regular merge candidate list. The pattern of spatial merge candidates is shown in Fig. 18. The distances between non-adjacent spatial candidates and current coding block are based on the width and height of current coding block. In Fig. 18, each small square corresponds to a NAMVP candidate and the candidates are ordered (as shown by the number inside the square) according to the distance. The line buffer restriction is not applied. In other words, the NAMVP candidates far away from a current block may have to be stored that may require a large buffer.

[0094] In the present invention, methods and apparatus to improve the coding performance of cross-component prediction are disclosed.

BRIEF SUMMARY OF THE INVENTION

[0095] A method and apparatus for video coding using coding tools including one or more cross component models related modes are disclosed. According to the method for the decoder side, coded data associated with a current block comprising a first-colour block and a second-colour block are received, wherein the current block is coded using CCP (Cross-Component Prediction). Coded syntax data corresponding to one or more syntaxes related to transform coding or residual coding applied to the current block are parsed. The coded syntax data are decoded to generate said one or more syntaxes by using one or more context models according to information associated with the CCP. The current block is decoded using the CCP, wherein said decoding comprises the transform coding of the current block or the residual coding of the current block using said one or more syntaxes.

[0096] In one embodiment, said one or more syntaxes are from a group comprising root_cbf, cb_cbf, cr_cbf, transform_skip_flag, lfnst_idx, mts_idx, tu_joint_cbcr_residual_flag, last_significant_position, cu_qp_delta_abs, cu_qp_delta_sign_flag, cu_chroma_qp_offset_flag, cu_chroma_qp_offset_idx, or a combination thereof.

[0097] In one embodiment, said one or more context models correspond to multiple context models and selection among the multiple context models depends on CCP type of the current block. In another embodiment, selection among the multiple context models depends on a prediction mode of a neighbouring block of the current block. For example, the prediction mode of the neighbouring block corresponds to one CCP mode. In another embodiment, selection among the multiple context models depends on a number of neighbouring blocks coded in a CCP mode.

[0098] In one embodiment, said one or more context models correspond to CCP single model, CCP multiple model, CCCM (Convolutional Cross-Component Mode), GLM (Gradient Linear Model), or a combination thereof.

[0099] The corresponding method for the encoder side is also disclosed.

[0100] According to another method, the current block is encoded or decoded by using the CCP, wherein said encoding or decoding uses information comprising a chroma syntax corresponding to one or more coded block flags comprising a root chroma block flag.

[0101] In one embodiment, an inherited CCP candidate from a neighbouring block is included in a merge candidate list for the current block and a merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not.

[0102] In one embodiment, said one or more coded block flags further comprise two or more component coded block flags corresponding to said two or more chroma blocks. In one embodiment, if the root chroma coded block flag is false, each of said two or more component coded block flags corresponding to said two or more chroma blocks is equal to 0, or if the root chroma coded block flag is true, at least one of said two or more component coded block flags corresponding to said two or more chroma blocks is equal to 1.

[0103] In one embodiment, said one or more coded block flags further comprise two or more component coded block flags corresponding to said two or more chroma blocks. In one embodiment, if the root chroma coded block flag is false, each said two or more component coded block flags corresponding to the said two or more chroma blocks is equal to 0, or if the root chroma coded block flag is true, at least one of said two or more component coded block flags corresponding to said two or more chroma blocks is equal to 1.

[0104] According to yet another method, a merge candidate list is generated for the current block, wherein the merge candidate list includes an inherited CCP candidate from a neighbouring block. A merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not is signalled or parsed. A skip flag is signalled or parsed before the merge flag, wherein when the skip flag is true, the merge flag is inferred. The current block is encoded or decoded by using information comprising the merge candidate list.

[0105] In one embodiment, when the skip flag is true, an index associated with the inherited CCP candidate is signalled or parsed. In one embodiment, when the merge flag is inferred to be 1, the current block inherits said one or more cross-component model parameters from the inherited CCP candidate. In one embodiment, when the merge flag is inferred to be 0, a CCP mode index is signalled or parsed.

[0106] In one embodiment, a high-level syntax is signalled or parsed in SPS (Sequence Parameter Set), PPS (Picture Parameter Set), PH (Picture header) or SH (Slice header) to indicate whether the skip flag is allowed for a current sequence, picture, or slice.

BRIEF DESCRIPTION OF THE DRAWINGS

[0107] Fig. 1A illustrates an exemplary adaptive Inter/Intra video coding system incorporating loop processing.

[0108] Fig. 1B illustrates a corresponding decoder for the encoder in Fig. 1A.

[0109] Fig. 2 illustrates examples of a multi-type tree structure corresponding to vertical binary splitting (SPLIT_BT_VER), horizontal binary splitting (SPLIT_BT_HOR), vertical ternary splitting (SPLIT_TT_VER), and horizontal ternary splitting (SPLIT_TT_HOR).

[0110] Fig. 3 illustrates an example of the signalling mechanism of the partition splitting information in quadtree with nested multi-type tree coding tree structure.

[0111] Fig. 4 shows an example of a CTU divided into multiple CUs with a quadtree and nested multi-type tree coding block structure, where the bold block edges represent quadtree partitioning and the remaining edges represent multi-type tree partitioning.

[0112] Fig. 5 shows some examples of TT split forbidden when either width or height of a luma coding block is larger than 64.

[0113] Fig. 6 shows the intra prediction modes as adopted by the VVC video coding standard.

[0114] Figs. 7A-B illustrate examples of wide-angle intra prediction a block with width larger than height (Fig. 7A) and a block with height larger than width (Fig. 7B).

[0115] Fig. 8 shows an example of the location of the left and above samples and the sample of the current block involved in the CCLM_LT mode.

[0116] Fig. 9 shows an example of classifying the neighbouring samples into two groups.

[0117] Fig. 10A illustrates an example of the CCLM model.

[0118] Fig. 10B illustrates an example of the effect of the slope adjustment parameter "u" for model update.

[0119] Fig. 11 illustrates an example of spatial part of the convolutional filter.

[0120] Fig. 12 illustrates an example of reference area with paddings used to derive the filter coefficients.

[0121] Fig. 13 illustrates the 4 gradient patterns for Gradient Linear Model (GLM).

[0122] Fig. 14 illustrates the neighbouring blocks used for deriving spatial merge candidates for VVC.

[0123] Fig. 15 illustrates the possible candidate pairs considered for redundancy check in VVC.

[0124] Fig. 16 illustrates an example of temporal candidate derivation, where a scaled motion vector is derived according to POC (Picture Order Count) distances.

[0125] Fig. 17 illustrate the positions for the temporal candidate selected between candidates C_0 and C_1.

[0126] Fig. 18 illustrates an exemplary pattern of the non-adjacent spatial merge candidates.

[0127] Fig. 19 illustrates examples of CCM information propagation, where the blocks with dash line (i.e., A, E, G) are coded in cross-component mode (e.g., CCLM, MMLM, GLM, CCCM).

[0128] Fig. 20 illustrates an example of inheriting temporal neighbouring model parameters.

[0129] Figs. 21A-B illustrates two search patterns for inheriting non-adjacent spatial neighbouring models.

[0130] Figs. 22A-B illustrate examples for constructing the history table of the current region from the history table of the region having the same beginning geometric position of the current region (Fig. 22A) or from the history table of the region containing the centre geometric position of the current region (Fig. 22B).

[0131] Fig. 23 illustrates an example to map motion information for the to-be referenced positions in a non-available region to pre-defined positions, where the pre-defined positions are located at one line above the above-first CTU row.

[0132] Fig. 24 illustrates an example to map motion information for the to-be referenced positions in a non-available region to pre-defined positions, where the pre-defined positions are located at the bottom line of respective CTU rows.

[0133] Fig. 25 illustrates an example to map motion information for the to-be referenced positions in a non-available region to pre-defined positions, where the pre-defined positions are located at the bottom line or the centre line of respective CTU rows.

[0134] Fig. 26 illustrates an example to map motion information for the to-be referenced positions in a non-available region to pre-defined positions, where the pre-defined positions are located at the bottom line of respective CTU rows or one CTU row above the respective CTU rows.

[0135] Fig. 27 illustrates an example of neighbouring templates for calculating model error.

[0136] Fig. 28 illustrates an example of inheriting candidates from the candidates in the candidate list of neighbour blocks.

[0137] Fig. 29 illustrates an example of subsampling the inter coding or CCM information at the left-above position of each 2x2 grids in the CTU-level buffer before the information is saved to the picture-level buffer.

[0138] Fig. 30 illustrates one embodiment of the present invention, where if the skip flag is true, the merge flag will be inferred to 1, and the CBF of the Cb component and the CBF of the Cr component are both inferred to 0.

[0139] Fig. 31 illustrates one embodiment of the present invention, where if the skip flag is true, the merge flag will be inferred to 1 and the CBF of the Cb component and the CBF of the Cr component are both inferred to 0, and the merge flag is inferred to 0 and the CBFs of the Cb and Cr components both are inferred to 0.

[0140] Fig. 32 illustrates a flowchart of an exemplary video decoding system that decodes syntax data by using one or more context models according to information associated with the cross-component prediction according to an embodiment of the present invention.

[0141] Fig. 33 illustrates a flowchart of an exemplary video encoding system that encodes syntax data by using one or more context models according to information associated with the cross-component prediction according to an embodiment of the present invention.

[0142] Fig. 34 illustrates a flowchart of an exemplary video coding system that encodes or decoded a current block uses information comprising a chroma syntax corresponding to a coded block flag shared by said two or more chroma blocks according to an embodiment of the present invention.

[0143] Fig. 35 illustrates a flowchart of an exemplary video coding system that uses a merge flag to indicate whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0144] It will be readily understood that the components of the present invention, as generally described and illustrated in the figures herein, may be arranged and designed in a wide variety of different configurations. Thus, the following more detailed description of the embodiments of the systems and methods of the present invention, as represented in the figures, is not intended to limit the scope of the invention, as claimed, but is merely representative of selected embodiments of the invention. References throughout this specification to “one embodiment,” “an embodiment,” or similar language mean that a particular feature, structure, or characteristic described in connection with the embodiment may be included in at least one embodiment of the present invention. Thus, appearances of the phrases “in one embodiment” or “in an embodiment” in various places throughout this specification are not necessarily all referring to the same embodiment.

[0145] Furthermore, the described features, structures, or characteristics may be combined in any suitable manner in one or more embodiments. One skilled in the relevant art will recognize, however, that the invention can be practiced without one or more of the specific details, or with other methods, components, etc. In other instances, well-known structures, or operations are not shown or described in detail to avoid obscuring aspects of the invention. The illustrated embodiments of the invention will be best understood by reference to the drawings, wherein like parts are designated by like numerals throughout. The following description is intended only by way of example, and simply illustrates certain selected embodiments of apparatus and methods that are consistent with the invention as claimed herein.

[0146] In order to improve the coding performance of cross-component prediction, various schemes are disclosed.

[0147] In ECM, various methods of cross-component prediction (CCP) are proposed to improve the intra chroma coding efficiency. For example,

- CCLM_L, CCLM_T, CCLM_LT, MMLM_L, MMLM_T, MMLM_LT
- CCCM_L, CCCM_T, CCCM_LT, MMCCCM_L, MMCCCM_T, MMCCCM_LT
- GLM_L, GLM_T, GLM_LT, MMGLM_L, MMGLM_T, MMGLM_LT
- GLM2_L, GLM2_T, GLM2_LT, MMGLM2_L, MMGLM2_T, MMGLM2_LT (GLM2 is the GLM with the corresponding luma term)
- For each GLM mode, 4 gradient kernels can be selected.

[0148] With the help of these cross-component models, intra chroma prediction becomes more accurate, and the entropy of residue is also getting smaller. Based on this observation, this invention proposes the following methods to improve the coding performance of intra chroma prediction:

[0149] **Improve CBF Coding**

[0150] In previous coding standards, since intra coding usually cannot predict perfectly, root_cbf (i.e., indicating if any transform blocks of all colour components have transform coefficient) is not signalled when current block is coded by intra mode. However, by using various cross-component models which utilize the luma reconstructed samples as model input, chroma component can be predicted more accurately, and the probability that all the CBFs (Coding Block Flag) or coded flags of chroma components in the intra coding mode are equal to 0 is increasing. Various methods are proposed to improve the CBF coding efficiency based on this assumption.

[0151] In one embodiment, root_cbf syntax can be used for intra coded blocks to indicate whether any of transform block of each colour component has any transform coefficient.

[0152] In one embodiment, a new flag, root_chroma_cbf is proposed, which is used to indicate whether any transform block of chroma component has any transform coefficient. For example, if the root_chroma_cbf flag is true, it means that at least one of the Cb or Cr component has a CBF flag equal to 1. Therefore, when the CBF of the Cb component is 0, the CBF of the Cr component is implicitly inferred to be 1 and is not signalled. Otherwise, if root_chroma_cbf flag is false, it means that both the CBF of the Cb component and the CBF of the Cr component are equal to 0.

[0153] **Improving the Transform Coding and Residual Coding by Adding Context Model for Intra Chroma CCP Modes and Inherited Cross-Component Models**

[0154] Since the intra chroma prediction using CCP modes can be more accurate than using non-CCP modes and the inherited cross-component model is also a CCP mode, the syntax in the transform coding and the residual coding can be improved by adding one or more context models according to CCP mode related information and/or additional context models depending on whether the current block uses an inherited cross-component model or not.

[0155] In one embodiment, the syntax associated with the transform coding and the residual coding can include, but not limited to root_cbf, CBF of the Cb component, CBF of the Cr component, transform_skip_flag, lfnst_idx, mts_idx, tu_joint_cbcr_residual_flag, last_significant_position, cu_qp_delta_abs, cu_qp_delta_sign_flag, cu_chroma_qp_offset_flag, and/or cu_chroma_qp_offset_idx.

[0156] In one embodiment, all syntax in the transform coding and the residual coding can have one or more context models according to CCP mode related information. In another embodiment, part of syntax in the transform coding and the residual coding can have one or more context models according to CCP mode related information. In another embodiment, all syntax elements in the transform coding and the residual coding can have additional context models depending on whether the current block uses an inherited cross-component model or not.

[0157] In another embodiment, part of syntax elements in the transform coding and the residual coding can have one or more context models depending on whether the current block uses an inherited cross-component model or not.

[0158] In one embodiment, when the current block is coded by intra mode and the intra prediction mode is one of all CCP modes or an inherited cross-component model, the additional context model for the syntax associated with the transform coding and the residual coding is used.

[0159] In another embodiment, when current block is coded by intra mode and the intra prediction mode is one of a sub-group of all CCP modes, the additional context model for the syntax associated with the transform coding and the residual coding is used.

[0160] In another embodiment, there are multiple additional context models, and the decision of context model can depend on the CCP type of the current coded block. For example, the first additional context model is used if the current block is coded by CCP single model, and the second additional context model is used if the current block is coded by CCP multi-model. For another example, the first additional context model is used if the current block is coded by LM model, the second additional context model is used if the current block is coded by CCCM model, and the third additional context model is used if the current block is coded by GLM model.

[0161] In another embodiment, there are multiple additional context models, and the decision of context model for the syntax associated with the transform coding and the residual coding can be related to the prediction mode of neighbouring coded block. For example, the first additional context model is used if only the above block is coded by CCP mode, the second additional context model is used if only the left block is coded by CCP mode and the third additional context model is used if both the above block and the left block are coded by CCP mode.

[0162] In another embodiment, there are multiple additional context models, and the decision of context model can depend on the number of neighbouring blocks coded by CCP mode. For example, the first additional context model is used if the number of neighbouring blocks coded by CCP mode is equal to the first number. The second additional context model is used if the number of neighbouring blocks coded by CCP mode is equal to the second number, and so on.

[0163] In order to improve the prediction accuracy or coding performance of cross-component prediction, various schemes related to inheriting cross-component models are disclosed.

[0164] **Guided Parameter Set for Refining the Cross-Component Model Parameters**

[0165] According to this method, the guided parameter set is used to refine the derived model parameters by a specified CCLM mode. For example, the guided parameter set is explicitly signalled in the bitstream, after deriving the model parameters, the guided parameter set is added to the derived model parameters as the final model parameters. The guided parameter set contain at least one of a differential scaling parameter (dA), a differential offset parameter (dB), and a differential shift parameter (dS). For example, equation (1) can be rewritten as:

$$\mathrm{pred_C}(i,j) = \left(\left(\alpha' \cdot \mathrm{rec_L}'(i,j)\right) \gg s \right) + \beta,$$

[0166] and if dA is signalled, the final prediction is:

$$\mathrm{pred_C}(i,j) = \left(\left((\alpha' + dA) \cdot \mathrm{rec_L}'(i,j)\right) \gg s \right) + \beta.$$

[0167] Similarly, if dB is signalled, then the final prediction is:

$$\mathrm{pred_C}(i,j) = \left(\left(\alpha' \cdot \mathrm{rec_L}'(i,j)\right) \gg s \right) + (\beta + dB).$$

[0168] If dS is signalled, then the final prediction is:

$$\mathrm{pred_C}(i,j) = \left(\left(\alpha' \cdot \mathrm{rec_L}'(i,j)\right) \gg (s + dS) \right) + \beta.$$

[0169] If dA and dB are signalled, then the final prediction is:

$$\mathrm{pred_C}(i,j) = \left(\left((\alpha' + dA) \cdot \mathrm{rec_L}'(i,j)\right) \gg s \right) + (\beta + dB).$$

[0170] The guided parameter set can be signalled per colour component. For example, one guided parameter set is signalled for Cb component, and another guided parameter set is signalled for Cr component. Alternatively, one guided parameter set can be signalled and shared among colour components. The signalled dA and dB can be a positive or negative value. When signalling dA, one bin is signalled to indicate the sign of dA. Similarly, when signalling dB, one bin is signalled to indicate the sign of dB.

[0171] For another embodiment, dA and dB can be the LSB (Least Significant Bits) part of the final scaling and offset parameters. For example, if *m* bits are required to represent the final scaling parameters, then dA is the LSB part of the final scaling parameters, and *n* bits ($m > n$) are used to represent dA, where the MSB part (*m-n* bits) of the final scaling parameters are implicitly derived. In other words, for the final scaling parameters, the MSB part of the final scaling parameters is taken from the MSB part of α', and the LSB part of the final scaling parameters is from the signalled dA. Similarly, if *p* bits are required to represent the final offset parameters, dB is the LSB of the final offset parameters, and *q* bits ($p > q$) are used to represent dB, where the MSB part (*p-q* bits) of the final offset parameters are implicitly derived. In other words, for the final offset parameters, the MSB part of the final offset parameters is taken from the MSB part of β, and the LSB part of the final offset parameters is from the signalled dB.

[0172] For another embodiment, if dA is signalled, dB can be implicitly derived from the average value of neighbouring (e.g. L-shape) reconstructed samples. For example, in VVC, four neighbouring luma and chroma reconstructed samples are selected to derived model parameters. Suppose the average value of neighbouring luma and chroma samples are lumaAvg and chromaAvg, then β is derived by $\beta = \text{chromaAvg} - (\alpha' + \text{dA}) \cdot \text{lumaAvg}$. The average value of neighbouring luma samples (i.e., lumaAvg) can be calculated by all selected luma samples, the luma DC mode value of the current luma CB, or the average of the maximum and minimum luma samples (e.g., $\text{lumaAvg} = (\text{Max}(x_A^0, x_A^1) + \text{Min}(x_B^0, x_B^1) + 1) \gg 1$, or $\text{lumaAvg} = (\text{Min}(x_A^0, x_A^1) + \text{Max}(x_B^0, x_B^1) + 1) \gg 1$). Similarly, average value of neighbouring chroma samples (i.e., chromaAvg) can be calculated by all selected chroma samples, the chroma DC mode value of the current chroma CB, or the average of the maximum and minimum chroma samples (e.g., $\text{chromaAvg} = (\text{Max}(y_A^0, y_A^1) + \text{Min}(y_B^0, y_B^1) + 1) \gg 1$, or $\text{chromaAvg} = (\text{Min}(y_A^0, y_A^1) + \text{Max}(y_B^0, y_B^1) + 1) \gg 1$). Note, for non-4:4:4 colour subsampling format, the selected neighbouring luma reconstructed samples can be from the output of CCLM downsampling process.

[0173] For another embodiment, the shift parameter, *s*, can be a constant value (e.g., *s* can be 3, 4, 5, 6, 7, or 8), and dS is equal to 0 and no need to be signalled.

[0174] For another embodiment, in MMLM, the guided parameter set can also be signalled per model. For example, one guided parameter set is signalled for one model and another guided parameter set is signalled for another model. Alternatively, one guided parameter set is signalled and shared among linear models. Or only one guided parameter set is signalled for one selected model, and another model is not further refined by guided parameter set.

[0175] In another embodiment, the MSB part of α' is selected according to the costs of possible final scaling parameters. That is, one possible final scaling parameter is derived according to the signalled dA and one possible value of MSB for α'. For each possible final scaling parameter, the cost defined by the sum of absolute difference between neighbouring reconstructed chroma samples and corresponding chroma values generated by the CCLM model with the possible final scaling parameter is calculated and the final scaling parameter is the one with the minimum cost. In one embodiment, the cost function is defined as the summation of square error.

[0176] **Inherit Neighbouring Model Parameters for Refining the Cross-Component Model Parameters**

[0177] The final scaling parameter of the current block is inherited from the neighbouring blocks and further refined by dA (e.g., dA derivation or signalling can be similar or the same as the method in the previous "Guided parameter set for refining the cross-component model parameters"). Once the final scaling parameter is determined, the offset parameter (e.g., β in CCLM) is derived based on the inherited scaling parameter and the average value of neighbouring luma and chroma samples of the current block. For example, if the final scaling parameter is inherited from a selected neighbouring block, and the inherited scaling parameter is α'_{nei}, then the final scaling parameter is (α'_{nei} + dA). For yet another embodiment, the final scaling parameter is inherited from a historical list and further refined by dA. For example, the historical list records the most recent j entries of final scaling parameters from previous CCLM-coded blocks. Then, the final scaling parameter is inherited from one selected entry of the historical list, α'_{list}, and the final scaling parameter is (α'_{list} + dA). For yet another embodiment, the final scaling parameter is inherited from a historical list or the neighbouring blocks, but only the MSB (Most Significant Bit) part of the inherited scaling parameter is taken, and the LSB (Least Significant Bit) of the final scaling parameter is from dA. For yet another embodiment, the final scaling parameter is inherited from a historical list or the neighbouring blocks, but does not further refine by dA.

[0178] For yet another embodiment, after inheriting model parameters, the offset can be further refined by dB. For example, if the final offset parameter is inherited from a selected neighbouring block, and the inherited offset parameter is β'_{nei}, then the final scaling parameter is (β'_{nei} + dB). For still another embodiment, the final offset parameter is inherited from a historical list and further refined by dB. For example, the historical list records the most recent j entries of final scaling parameters from previous CCLM-coded blocks. Then, the final scaling parameter is inherited from one selected entry of the historical list, β'_{list}, and the final scaling parameter is (β'_{list} + dB).

[0179] For yet another embodiment, if the inherited neighbour block is coded with CCCM, the filter coefficients (c_i) are inherited. The offset parameter (e.g., $c_6 \times B$ or c_6 in CCCM) can be re-derived based on the inherited parameter and the average value of neighbouring corresponding position luma and chroma samples of the current block. For still another embodiment, only partial filter coefficients are inherited (e.g., only n out of 6 filter coefficients are inherited, where $1 \le n < 6$), the rest filter coefficients are further re-derived using the neighbouring luma and chroma samples of the current block.

[0180] For still another embodiment, if the inherited candidate applies GLM gradient pattern to its luma reconstructed samples, the current block shall also inherit the GLM gradient pattern of the candidate and apply to the current luma reconstructed samples.

[0181] For still another embodiment, if the inherited neighbour block is coded with multiple cross-component models (e.g., MMLM, or CCCM with multi-model), the classification threshold is also inherited to classify the neighbouring samples of the current block into multiple groups, and the inherited multiple cross-component model parameters are further assigned to each group. For yet another embodiment, the classification threshold is the average value of the neighbouring reconstructed luma samples, and the inherited multiple cross-component model parameters are further assigned to each group. Similarly, once the final scaling parameter of each group is determined, the offset parameter of each group is re-derived based on the inherited scaling parameter and the average value of neighbouring luma and chroma samples of each group of the current block. For another example, if CCCM with multi-model is used, once the final coefficient parameter of each group is determined (e.g., c_0 to c_5 except for c_6 in CCCM), the offset parameter (e.g., $c_6 \times B$ or c_6 in CCCM) of each group is re-derived based on the inherited coefficient parameter and the neighbouring luma and chroma samples of each group of the current block.

[0182] For still another embodiment, inheriting model parameters may depend on the colour component. For example, Cb and Cr components may inherit model parameters or model derivation method from the same candidate or different candidates. For yet another example, only one of colour components inherits model parameters, and the other colour component derives model parameters based on the inherited model derivation method (e.g., if the inherit candidate is coded by MMLM or CCCM, the current block also derives model parameters based on MMLM or CCCM using the current neighbouring reconstructed samples). For still another example, only one of colour components inherits model parameters, and the other colour component derives its model parameters using the current neighbouring reconstructed samples.

[0183] For still another example, if Cb and Cr components can inherit model parameters or model derivation method from different candidates. The inherited model of Cr can depend on the inherited model of Cb. For example, possible cases include but not limited to (1) if the inherited model of Cb is CCCM, the inherited model of Cr shall be CCCM; (2) if the inherit model of Cb is CCLM, the inherit model of Cr shall be CCLM; (3) if the inherited model of Cb is MMLM, the inherited model of Cr shall be MMLM; (4) if the inherited model of Cb is CCLM, the inherited model of Cr shall be CCLM or MMLM; (5) if the inherited model of Cb is MMLM, the inherited model of Cr shall be CCLM or MMLM; (6) if the inherited model of Cb is GLM, the inherited model of Cr shall be GLM.

[0184] For yet another embodiment, after decoding a block, the (CCM) information cross-component model of the current block is derived and stored for later reconstruction process of neighbouring blocks using inherited neighbours model parameter. The CCM information mentioned in this disclosure includes but not limited to prediction mode (e.g., CCLM, MMLM, CCCM), GLM pattern index, model parameters, or classification threshold. For example, even the current block is coded by inter prediction, the cross-component model parameters of the current block can be derived by using the current luma and chroma reconstructed or predicted samples. Later, if another block is predicted by using inherited neighbours model parameters, it can inherit the model parameters from the current block. For another example, the current block is coded by cross-component prediction, the cross-component model parameters of the current block are re-derived by using the current luma and chroma reconstructed or predicted samples. For another example, the stored cross-component model can be CCCM, LM_LA (i.e., single model LM using both above and left neighbouring samples to derive model), or MMLM_LA (multi-model LM using both above and left neighbouring samples to derive model). For still example, even the current block is coded by non-cross-component intra prediction (e.g., DC, planar, intra angular modes, MIP, or ISP), the cross-component model parameters of the current block are derived by using the current luma and chroma reconstructed or predicted samples. For another example, even the current block is coded by cross-component prediction, the cross-component model parameters of the current block are re-derived by using the current luma and chroma reconstructed or predicted samples. Later the re-derived model parameters are combined with the original cross-component models which is used in reconstructing the current block. For combining with the original cross-component models, it can use the model combination methods mentioned in sections entitled: Models Generated Based on Other Inherited Models and entitled: Inheriting Multiple Cross-Component Models. For example, assume the original cross-component model parameters are $\{c_0^0, c_1^0, \ldots, c_{M-1}^0\}$, and the re-derived cross-component model parameters are $\{c_0^1, c_1^1, \ldots, c_{M-1}^1\}$. The final cross-component model is $\{(1-\alpha) \times c_0^0 + \alpha \times c_0^1, (1-\alpha) \times c_1^0 + \alpha \times c_1^1, \ldots, (1-\alpha) \times c_{M-1}^0 + \alpha \times c_{M-1}^1\}$, where α is a weighting factor which can be predefined or implicitly derived by neighbouring template cost.

[0185] For another embodiment, when inheriting a cross-component model from a neighbour merge candidate that was coded by a cross-component mode (e.g. CCLM and CCCM, …), a flag can be signalled to indicate/select if the re-derived model is used. If the flag is 0, the cross-component model used to encode the neighbour merge candidate is inherited. If the flag is 1, the cross-component model re-derived based on the luma and chroma reconstructed or predicted samples of the neighbour merge candidate is inherited.

[0186] For still another example, when the current slice is a non-intra slice (e.g., P slice or B slice), a cross-component model of the current block is derived and stored for later reconstruction process of neighbouring blocks using inherited neighbours model parameter. For still another embodiment, when the current block is inter-coded, the CCM information of the current inter-coded block is derived by copying the CCM information from its reference block that has CCM information in a reference picture, located by the motion information of the current inter-coded block. For example, as shown in Fig. 19, the block B in a P/B picture 1920 is inter-coded, then the CCM information of block B is obtained by copying CCM information from its referenced block A in an I picture 1910. It should be noted that the current block can also copy the CCM information from an intra-coded block in an P/B picture. For example, as shown in the Fig. 19, the block D in a P/B picture 1930 is inter-coded, then the CCM information of block D is obtained by copying CCM information from its referenced block E that is intra-coded in the P/B picture 1920. For still another embodiment, if the reference block in a reference picture is also inter-coded, the CCM information of the reference block is obtained by copying the CCM information from another reference block in another reference picture. For example, as shown in the Fig. 19, the current block C in a current P/B picture 1930 is inter-coded and its referenced block B is also inter-coded, due to the CCM information of block B is obtained by copying the CCM information from block A, then the CCM information of block A is also propagated to the current block C. For still another embodiment, when the current block is inter-coded with bi-directional prediction, if one of its reference blocks is intra-coded and has CCM information, the CCM information of the current block is obtained by copying the CCM information from its intra-coded reference block in a reference picture. For example, suppose block F is inter-coded with bi-prediction and has reference blocks G and H. Block G is intra-coded and has CCM information. The CCM information of block F is obtained by copying the CCM information from the block G coded in CCM mode. For still another embodiment, when the current block is inter-coded with bi-directional prediction, the CCM information of the current block is the combination of the CCM models of its reference blocks (as the method mentioned in section entitled: Inheriting Multiple Cross-Component Models).

[0187] When deriving cross-component models for the current block by using the current luma and chroma reconstructed or predicted samples, in one embodiment, if the current derived model error is greater than a threshold, the current derived model is discarded and not stored. For example, it can input the current luma reconstructed samples to the model, calculate the distortion between the model output and the current chroma reconstructed samples, and then the calculated distortion is normalized by the current block size or number of samples used in calculating the distortion. If the normalized distortion is greater than or equal to a threshold, the current derived model is discarded and not stored.

[0188] Whether to derive cross-component models for the current block or not can depend on the current block size or area. For example, for small blocks (e.g., block width/height less than or equal to a threshold, or block area less than or equal a threshold), it is not allowed to derive cross-component models. For another example, for large blocks (e.g., block width/height greater than or equal to a threshold, or block area greater than or equal a threshold), it is not allowed to derive cross-component models.

[0189] **Inherit CCM Information**

[0190] In one embodiment, the cross-component model (CCM) information of inherited cross-component model can be stored together with the inherited model parameters. As mentioned earlier in this disclosure, the CCM information includes, but not limited to prediction mode (e.g., CCLM, MMLM, CCCM), model index for indicating which model shape is used in convolutional model, classification threshold for multi-model, down-sampling filter flag, down-sampling filtering index, number of neighbouring lines used to derive model, types of templates used to derive model, post-filtering flag or model parameters.

[0191] In one embodiment, CCLM model can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a CCLM model.

[0192] In another embodiment, CCLM model with non-linear term can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a CCLM model with a non-linear term.

[0193] In one embodiment, CCCM model can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a CCCM model. The luma offset and the chroma offset used for adjusting the input of CCCM model can also be stored in the CCM information.

[0194] In another embodiment, CCCM model with different convolution filter shape can be inherited. In addition to model parameters and prediction mode, a CCCM mode index can also be stored in CCM information for indicating which convolution filter shape is used for the inherited CCCM model. For example, a CCCM model with different convolution filter shape can only contain the spatial terms in the horizontal direction. For another example, a CCCM model with different convolution filter shape can only contain the spatial terms in vertical direction. For another example, a CCCM model with different convolution filter shape can only contain the spatial terms in diagonal direction. For another example, a CCCM model with different convolution filter shape can only contain the spatial terms in anti-diagonal direction. For another example, a CCCM model with different convolution filter shape can contain the X-shape spatial terms.

[0195] In another embodiment, CCCM model using non-downsampled samples can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a CCCM model using non-downsampled samples.

[0196] In another embodiment, CCCM model with multiple down-sampling filter can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a CCCM model with multiple down-sampling filter, and a model index can be also stored in CCM information for indicating which variant of

CCCM model with multiple down-sampling filter is inherited.

[0197] In another embodiment, a mixed CCCM model consist of various terms (e.g. spatial term, gradient term, location term, non-linear term and bias term) can be inherited. The gradient term can be calculated in either the downsampled domain or the non-downsampled domain. The location term can be calculated with respect to the top-left coordinates of the current block or the picture. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a mixed CCCM model consist of various terms. If there are multiple types of mixed CCCM models, a model index can also be stored in the CCM information for indicating which type of mixed CCCM model is inherited. For example, gradient and location based CCCM (GL-CCCM) proposed in JVET-AB0119 (Ramin G. Youvalari, et al., "Non-EE2: Gradient and location based convolutional cross-component model (GL-CCCM) for intra prediction", Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 28th Meeting, Mainz, DE, 20–28 October 2022, Document: JVET- AB0119) is a mixed CCCM model which consist of one spatial term in centre position, two gradient terms for the horizontal direction and vertical direction, two location terms X and Y for the relative horizontal location and relative vertical location, one non-linear term and one bias term. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a GL-CCCM model.

[0198] In one embodiment, GLM model can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a GLM model, and a down-sampling filtering index can be also stored in CCM information for indicating which gradient down-sampled filter is used for the inherited GLM model.

[0199] In another embodiment, GLM model with luma term can be inherited. In addition to storing model parameters, a prediction mode can be stored in the CCM information for indicating that the inherited model is a GLM model with luma term, and a down-sampling filtering index can be also stored in CCM information for indicating which gradient down-sampled filter is used for the inherited GLM model with luma term.

[0200] In one embodiment, any kind of cross-component multi-model can be inherited. In addition to storing model parameters and prediction mode, a multi-model on/off flag can be stored in the CCM information for indicating whether the inherited CCM model is a multi-model or not. If the multi-model on/off flag is true, the multi-model classification threshold is also stored in the CCM information.

[0201] In one embodiment, the CCM information can include information to indicate how the inherited model is derived. For example, the CCM information can include the number of neighbouring lines used to derive the cross-component model and/or the types of templates used to derive model. For example, a set of templates can be used to derive a CCCM model. The set of templates include template with different positions, sizes, and shapes. The CCM information can store the index of the template that the inherited CCCM model was derived based on. For example, the inherited CCCM model can be derived based on a top-only template, or a left-only template or a left-and-top template. For another example, the inherited CCCM model can be derived based on a 6-line template or a 2-line template.

[0202] In one embodiment, a post-filter flag can be stored in the CCM information. This information describes how the inherited model is used in the block the inherited model is from. If the post-filter flag is on, this indicates a filter is applied to the prediction of the block the inherited model is from.

[0203] **Refinement of Inherited Model Parameters**

[0204] In one embodiment, the inherited model parameters can be further refined based on the inherited CCM information. The inherited CCM information can include how the inherited model is derived, such as the type of template and/or the number of neighbouring lines used to derive the model. The refined parameters are derived based on local information. The refinement process can follow how the inherited model was derived and use the same type of template and/or the same number of neighbouring lines. For example, if the inherited model is CCLM and was derived based on the left-only template (i.e., the inherited model being CCLM_L), the offset parameter β can be derived from the average value of neighbouring left-template reconstructed samples of the current block. For example, if the inherited model is CCLM and was derived based on top-only template (i.e., the inherited model being CCLM_T), the offset parameter β can be derived from the average value of neighbouring top-template reconstructed samples of the current block. For another example, if the inherited model is CCCM and was derived with 2-line template. The offset value (e.g. c_6 in CCCM) can be re-derived based on the 2-line template reconstructed samples of the current block. For another example, if the inherited model is a multi-model (MMLM, CCCM with multi-model) and was derived based on left-only template, the classification threshold can be re-derived based on the left-template reconstructed samples of the current block.

[0205] In one embodiment, the inherited model parameters are further refined with different types of templates and/or different numbers of lines, and the final model parameters are determined by the template cost. The template cost is computed by applying the candidate refined model parameters on the neighbouring template to predict the template samples and compute the differences (e.g. SAD or SATD) between the predicted and the reconstructed samples. For example, if the inherited model is CCLM, the refined offset parameter $\beta'_L, \beta'_T, \beta'_{LT}$ is derived using the left-template, top-template and left-top template of the reconstructed samples of the current block respectively. If the template cost of applying β'_L is the minimum among $\beta'_L, \beta'_T, \beta'_{LT}$, then β'_L is chosen as the final offset parameter.

[0206] In another embodiment, the inherited model parameters can be further refined by a pre-defined value. The template cost is used to determine if the inherited model parameters are further refined. The template cost is computed by applying the candidate refined model parameters on the neighbouring template to predict the template samples and compute the differences (e.g. SAD or SATD) between the predicted and the reconstructed samples. For example, for CCCM mode, for each inherited model parameter c_i, the value is refined by dc_i, and the template costs of applying $c_i + dc_i$ and c_i are compared to determine which is the final model parameter value.

[0207] **Inherit Spatial Neighbouring Model Parameters**

[0208] For another embodiment, the inherited model parameters can be from a block that is an immediate neighbouring block. The models from blocks at pre-defined positions are added into the candidate list in a pre-defined order. For example, the pre-defined positions can be the positions depicted in Fig. 14, and the pre-defined order can be B_0, A_0, B_1, A_1 and B_2, or A_0, B_0, B_1, A_1 and B_2.

[0209] For still another embodiment, the pre-defined positions include the positions at the immediate above (W >> 1) or ((W >> 1) – 1) position if W is greater than or equal to TH, and the positions at the immediate left (H >> 1) or ((H >> 1) – 1) position if H is greater than or equal to TH, where W and H are the width and height of the current block, TH is a threshold value which can be 4, 8, 16, 32, or 64.

[0210] For still another embodiment, the maximum number of inherited models from spatial neighbours are smaller than the number of pre-defined positions. For example, if the pre-defined positions are as depicted in Fig. 14, there are 5 pre-defined positions. If pre-defined order is B_0, A_0, B_1, A_1 and B_2, and the maximum number of inherited models from spatial neighbours is 4, the model from B2 is added into the candidate list only when one of preceding blocks is not available or is not coded in cross-component model.

[0211] **Inheriting Temporal Neighbouring Model Parameters**

[0212] For still another embodiment, if the current slice/picture is a non-intra slice/picture, the inherited model parameters can be from the block in the previous coded slices/pictures. For example, as shown in the Fig. 20, the current block position is at (x, y) and the block size is $w \times h$. The inherited model parameters can be from the block at position (x', y') , (x', y' + h/2) , (x' + w/2, y'), (x' + w/2, y' + h/2), (x' + w, y'), (x', y' + h), or (x' + w, y' + h) of the previous coded slices/picture, where x' = x + Δx and y' = y + Δy. In one embodiment, if the prediction mode of the current block is intra, Δx and Δy are set to 0. If the prediction mode of the current block is inter prediction, Δx and Δy are set to the horizontal and vertical motion vector of the current block. In another embodiment, if the current block is inter bi-prediction, Δx and Δy are set to the horizontal and vertical motion vectors in reference picture list 0. In still another embodiment, if the current block is inter bi-prediction, Δx and Δy are set to the horizontal and vertical motion vectors in reference picture list 1.

[0213] For still another embodiment, if the current block is inter bi-prediction, the inherited model parameters can be from the block in the previous coded slices/pictures in the reference lists. For example, if the horizontal and vertical motion vector in reference picture list 0 is Δx_{L0} and Δy_{L0}, the motion vector can be scaled to other reference pictures in the reference list 0 and 1. If the motion vector is scaled to the i^{th} reference picture in the reference list 0 as $(\Delta x_{L0,i0}, \Delta y_{L0,i0})$. The model can be from the block in the i^{th} reference picture in the reference list 0, and Δx and Δy are set to $(\Delta x_{L0,i0}, \Delta y_{L0,i0})$. For another example, if the horizontal and vertical motion vector in reference picture list 0 is Δx_{L0} and Δy_{L0} and the motion vector is scaled to the i^{th} reference picture in the reference list 1 as $(\Delta x_{L0,i1}, \Delta y_{L0,i1})$. The model can be from the block in the i^{th} reference picture in the reference list 1, and Δx and Δy are set to $(\Delta x_{L0,i1}, \Delta y_{L0,i1})$

[0214] **Inherit Non-Adjacent Spatial Neighbouring Models**

[0215] For another embodiment, the inherited model parameters can be from blocks that are spatial neighbouring blocks. The models from blocks at pre-defined positions are added into the candidate list in a pre-defined order. For example, the pattern of the positions and order can be as the pattern depicted in Fig. 18, where the distance between each position is the width and height of current coding block. For another embodiment, the distance between the positions that are closer to the current encoding block is smaller than the positions that are further away from the current block.

[0216] For still another embodiment, the maximum number of inherited models from non-adjacent spatial neighbours are smaller than the number of pre-defined positions. For example, if the pre-defined positions are as depicted in Figs. 21A-B, where two patterns (pattern 2110 in Fig. 21A and pattern 2120 in Fig. 21B) are shown. If the maximum number of inherited models from non-adjacent spatial neighbours is N, the search pattern 2 is used only when the number of available models from search pattern 1 is smaller than N.

[0217] **Inherit Model Parameters from History Table**

[0218] In one embodiment, the inherited model parameters can be from a cross-component model history table. The cross-component models in the history table can be added into the candidate list according to a pre-defined order. In one embodiment, the adding order of historical candidates can be from the beginning of the table to the end of the table. In another embodiment, the adding order of historical candidates can be from a certain pre-defined position to the end of the table. In another embodiment, the adding order of historical candidates can be from the end of the table to the beginning of the table. In another embodiment, the adding order of historical candidates can be from a certain pre-defined position to the beginning of the table. In another embodiment, the adding order of historical candidates can be in an interleaved manner (e.g., the first added is from the beginning of the table, the second added candidate from the end of the table and so on).

[0219] In one embodiment, single cross-component model history table can be maintained for storing the previous cross-component model, and the cross-component model history table can be reset at the start of the current picture, current slice, current tile, every M CTU rows or every N CTUs, where N and M can be any value greater than 0. In another embodiment, the cross-component model history table can be reset at the end of the current picture, current slice, current tile, current CTU row or current CTU.

[0220] In another embodiment, one picture can be divided into multiple regions, and for each region, a history table is kept. The history table 0 and one additional history table will be updated during the encoding/decoding process. The additional history table can be determined by the current position. For example, if the current CU is located in the second region, the additional history table to be updated is history table 2.

[0221] In another embodiment, multiple history tables are used for different updated frequencies. For example, the first history table is updated every CU, the second history table is updated every two CUs, the third history table is updated every four CUs and so on.

[0222] In another embodiment, multiple history tables are used for storing different types of cross-component models. For example, the first history table is used for storing single model, and the second history table is used for storing multi-model. For another example, the first history table is used for storing a gradient model, and the second history table is used for storing a non-gradient model. For another example, the first history table is used for storing a simple linear model (e.g., $y = ax + b$), and the second history table is used for storing a complicated model (e.g., CCCM).

[0223] In another embodiment, multiple history tables are used for different reconstructed luma intensity. For example, if the average of reconstructed luma samples in the current block is greater than a pre-defined threshold, the cross-component model will be stored in the first history table; otherwise, the cross-component model will be stores in the second history table. In another embodiment, multiple history tables are used for different reconstructed chroma intensities. For example, if the average of neighbouring reconstructed chroma samples in the current block is greater than a pre-defined threshold, the cross-component model will be stores in the first history table; otherwise, the cross-component model will be stored in the second history table.

[0224] In one embodiment, when adding historical candidates from multiple history tables to the candidate list, the adding order can be from the beginning of the certain table to the end of the certain table, and then add the next history table in the same order or in a reversed order. In another embodiment, the adding order can be from the end of the certain table to the beginning of the certain table, and then add the next history table in the same order or in a reversed order. In another embodiment, the adding order can be from the certain pre-defined position of the certain table to the end of the certain table, and then add the next history table in the same order or in a reversed order. In another embodiment, the adding order can be from the certain pre-defined position of the certain table to the beginning of the certain table, and then add the next history table in the same order or in a reversed order. In another embodiment, the adding order of historical candidates can be in an interleaved manner in a certain history table (e.g., the first added candidate from the beginning of the certain history table, the second added candidate from the end of the certain history table and so on), and then add the next history table in the same order or in a reversed order.

[0225] In another embodiment, the adding order can be from the beginning of each history table to the end of each history table. In another embodiment, the adding order can be from the end of each history table to the beginning of each history table. In another embodiment, the adding order can be from the certain pre-defined position of each history table to the end of each history table. In another embodiment, the adding order can be from the certain pre-defined position of each history table to the beginning of each history table. In another embodiment, the adding order of historical candidates can be in an interleaved manner in each certain history table (e.g., the first added candidates from the beginning of all history table, the second added candidates from the end of all history table and so on).

[0226] In one embodiment, multiple cross-component model history tables are used, but not all history tables will be used for creating the candidate list. Only history tables whose regions are close to the region of current block can be used to create the candidate list.

[0227] In one embodiment, if the historical candidates are used, the range for selecting non-adjacent candidates can be reduced by using a smaller distance between each position of non-adjacent candidate. In another embodiment, if the historical candidates are used, the number of non-adjacent candidates can be reduced by measuring the distance from the left-top position of the current block to the candidate position, and then exclude the candidate with the distance greater than a pre-defined threshold. In another embodiment, if the historical candidates are used, the number of non-adjacent candidates can be reduced by skipping the candidates that are not located in the same region. In another embodiment, if the historical candidates are used, the number of non-adjacent candidates can be reduced by skipping the candidates that are not located in the neighbouring regions. The range of neighbouring regions is pre-defined, and it can be M by N regions, where M and N can be any value greater than 0. In another embodiment, if the historical candidates are used, the range for selecting non-adjacent candidates can be reduced by skipping the second search pattern.

[0228] In another embodiment, one picture can be divided into multiple regions, and at least one history table is kept in each region. For a region of the current picture, it can use or combine the history tables of one or multiple regions in the previous coded pictures as the initial history table. For example, if a picture is divided into N regions, it can implicitly or explicitly select the history table from one of N regions in the previous coded pictures as the initial history table. The index of one of N regions can be signalled or implicitly derived from the corresponding region in the previous coded pictures. As shown in Figs. 22A-B, where the current picture 2220 is a P/B coded picture and the previous picture 2210 is an Intra coded picture. Each picture is divided into 4 regions as shown in 4 rectangular boxes. According to an embodiment of the present invention, the corresponding region in the previous coded pictures can be the region 2212 having the same beginning geometric position as the current region 2222 as shown in Fig. 22A or containing the centre geometric position of the current region 2222 as shown in Fig. 22B. For another example, it can combine more than one history tables in the previous coded regions/pictures to construct the history table of the current region (e.g., the method in the section entitled: Inheriting Candidates from the Candidates in the Candidate List of Neighbours).

[0229] **The Available Region of Non-Adjacent Spatial Candidates**

[0230] To limit the requirement buffer/storage resource, the available range for including non-adjacent spatial candidates should be constrained. In one embodiment, only the cross-component model (CCM) information in the current CTU can be referenced by the non-adjacent spatial candidate. In another embodiment, only the CCM information in the current CTU or left M CTUs can be referenced by the non-adjacent spatial candidate. M can be any integer larger than 0. In another embodiment, only the CCM information in the current CTU row can be referenced by the non-adjacent spatial candidate. In another embodiment, only the to-be referenced position within the current CTU row or above N CTU rows can be referenced. N can be any integer larger than 0. Note, the CCM information mentioned in this disclosure includes but not limited to prediction mode (e.g., CCLM, MMLM, CCCM), GLM pattern index, model parameters, or classification threshold.

[0231] In another embodiment, the CCM information in the current CTU, the current CTU row, the current CTU row + above N CTU rows, the current CTU + left M CTUs, or the current CTU + above N CTU rows + left M CTUs can be referenced without limits. Furthermore, the CCM information in other regions can only be referenced by a larger pre-defined unit. For example, the CCM information in the current CTU row is stored within a 4x4 grid, and for other CCM information outside the current CTU row is stored within a 16x16 grid. In other words, one 16x16 region only needs to store one CCM information, so the to-be referenced position shall be rounded to the 16x16 grid, or changed to the nearest position of 16x16 grid.

[0232] In another embodiment, the CCM information in the current CTU row, or the current CTU row + M CTU rows can be referenced without limits, and for the to-be referenced positions in the above CTU row, the positions will be mapped to one line above of current CTU, or the current CTU row + M CTU rows for referencing. This design can preserve most of the coding efficiency and doesn’t increase buffer by much for storing the CCM information of above CTU rows. For example, the CCM information in the current CTU row (2310) and the first CTU row above (2312) can be referenced without limits; and for the to-be referenced positions in the above-second (2320), above-third (2322), above-fourth CTU row, and so on, the positions will be mapped to one line (2330) above the above-first CTU row (as shown in Fig. 23). In Fig. 23, a dark circle indicates a non-available candidate 2340, a dark circle indicates a non-available candidate 2340, a dot-filled circle indicates an available candidate 2342 and an empty circle indicates a mapped candidate 2344. For example, the non-available candidate 2350 in the above-third (2322) CTU row is mapped to an available candidate

2352 in one line (2330) above the above-first CTU row (2312).

[0233] In the above example, the region that can be referenced without limits is close to the current CTU (e.g. the current CTU row or the above-first CTU row). However, the region according to the present invention is not limited to the exemplary region shown above. The region can be larger or smaller than the example shown above. In general, the region can be limited to be within one or more pre-define distances in a vertical direction, a horizontal direction or both from the current CTU. In the above example, the region is limited to 1 CTU height in the above vertical direction, which can be extended to 2 or 3 CTU heights if desired. In the case that left M CTUs are used, the limit is M CTU width for the current CTU row. The horizontal position of a to-be referenced position and the horizontal position of a mapped pre-defined position can be the same (e.g. position 2350 and position 2352 in the same horizontal position). However, other horizontal position may also be used.

[0234] In another embodiment, the CCM information in the current CTU row, or the current CTU row + M CTU rows can be referenced without limits. Furthermore, for the to-be referenced positions in the above CTU row, the positions will be mapped to the last line of the corresponding CTU row for referencing. For example, as shown in Fig. 24, the CCM information in the current CTU row (2310) and the first CTU row (2312) above can be referenced without limits, and for the to-be referenced positions in the above-second CTU row (2320), the positions will be mapped to the bottom line (2330) of the above-second CTU row (2320). For the to-be referenced positions in above third CTU row (2322), the positions will be mapped to the bottom line (2420) of the above-third CTU row (2322). For example, the non-available candidate 2350 in the above-third CTU row (2322) is mapped to a mapped candidate 2430 in the bottom line (2420) of the above-third CTU row (2322). The legend for the candidate types (i.e., 2340, 2342 and 2344) of Fig. 24 is the same as that in Fig. 23. In this example, the unconstrained region may include one or more above CTU rows (e.g., 1 CTU in Fig. 24). The above-second CTU row is above the unconstrained region. The above-third CTU row is also referred as an *above-above* CTU row since it is above the CTU row (i.e., the above-second CTU row) above the unconstrained region.

[0235] In another embodiment, the CCM information in the current CTU row, or the current CTU row + M CTU rows can be referenced without limits, and for the to-be referenced positions in above CTU row, the positions will be mapped to the last line or bottom line or centre line of the corresponding CTU row for referencing depending on the position of the to-be referenced CCM information. For example, as shown in Fig. 25, the CCM information in the current CTU row (2310) and the above-first CTU row (2312) can be referenced without limits, and for the to-be referenced position **1** in above-second CTU row (2320), the positions will be mapped to the bottom line (2330) of the above-second CTU row before referring. However, for the to-be referenced position **2** in above-second CTU row, the positions will be mapped to the centre line (2510) of the above-second CTU row (2320) before referring since it is closer to the centre line (2510) compared with bottom line

(2330). The legend for the candidate types (i.e., 2340, 2342 and 2344) of Fig. 25 is the same as that in Fig. 23.

[0236] In another embodiment, the CCM information in the current CTU row, or the current CTU row + M CTU rows can be referenced without limits, and for the to-be referenced positions in the above CTU row, the positions will be mapped to the last line or bottom line of the corresponding CTU row for referencing depending on the position of the to-be referenced CCM information. For example, as shown in Fig. 26, the CCM information in the current CTU row (2310) and the above-first CTU row (2312) can be referenced without limits, and for the to-be referenced position **1** in the above-second CTU row (2320), the positions will be mapped to the bottom line (2330) of the above-second CTU row (2320) before referring. However, for the to-be referenced position **2** in the above-second CTU row (2320), the positions will be mapped to the bottom line (2420) of the above-third CTU row (2322) before referring since it is closer to the bottom line (2420) of the above-third CTU row compared with bottom line (2330) of the above-second CTU row as shown in Fig. 26. The legend for the candidate types (i.e., 2340, 2342 and 2344) is the same as that in Fig. 23.

[0237] In another embodiment, the CCM information in the current CTU, or the current CTU + N left CTU can be referenced without limits, and for the left CTUs, the to-be referenced positions will be mapped to the very right line closest to the current CTU, or the current CTU + N left CTU. For example, the CCM information in the current CTU and first left CTU can be referenced without limits, and if the to-be referenced positions are in the second left CTU, the positions will be mapped to one line left to the first left CTU. If the to-be referenced positions are in the third left CTU, the positions will be mapped to one line left to first left CTU. For example, the CCM information in the current CTU and the first left CTU can be referenced without limits, and if the to-be referenced positions are in the second left CTU, the positions will be mapped to the very right line of the second left CTU. If the to-be referenced positions are in the third left CTU, the positions will be mapped to the very right line to the third left CTU.

[0238] In another embodiment, when the available range for including non-adjacent candidates is constrained, if the position of a non-adjacent candidate is outside of the available range, that candidate is skipped and will not be inserted into the candidate list. The available region can be the current CTU, current CTU row, current CTU row + above N CTU rows, current CTU + left M CTUs, or current CTU + above N CTU rows + left M CTUs.

[0239] **Models Generated Based on Other Inherited Models**

[0240] In another embodiment, a single cross-component model can be generated from a multiple cross-component model. For example, if a candidate is coded with multiple cross-component models (e.g., MMLM, or CCCM with multi-model), a single cross-component model can be generated by selecting the first or the second cross-component model in the multi cross-component models.

[0241] **Candidate List Construction**

[0242] In one embodiment, the candidate list is constructed by adding candidates in a pre-defined order until the maximum candidate number is reached. The candidates added may include all or some of the aforementioned candidates, but not limited to the aforementioned candidates. For example, the candidate list may include spatial neighbouring candidates, temporal neighbouring candidate, historical candidates, non-adjacent neighbouring candidates, single model candidates generated based on other inherited models or combined model (as mentioned later in section entitled: Inheriting Multiple Cross-Component Models). For another example, the candidate list can include the same candidates as previous example, but the candidates are added into the list in a different order.

[0243] In another embodiment, if all the pre-defined neighbouring and historical candidates are added but the maximum candidate number is not reached, some default candidates are added into the candidate list until the maximum candidate number is reached.

[0244] In one sub-embodiment, the default candidates include but not limited to the candidates described below. The final scaling parameter α is from the set $\{0, {}^1/_8, -{}^1/_8, +{}^2/_8, -{}^2/_8, +{}^3/_8, -{}^3/_8, +{}^4/_8, -{}^4/_8\}$, and the offset parameter $\beta = 1/(1 \ll bit\_depth)$ is derived based on neighbouring luma and chroma samples. For example, if the average value of neighbouring luma and chroma samples are lumaAvg and chromaAvg, then β is derived by $\beta = \mathrm{chromaAvg} - \alpha \cdot \mathrm{lumaAvg}$. The average value of neighbouring luma samples (lumaAvg) can be calculated by all selected luma samples, the luma DC mode value the current luma CB, or the average of the maximum and minimum luma samples (e.g., $\mathrm{lumaAvg} = (\mathrm{Max}(x_A^0, x_A^1) + \mathrm{Min}(x_B^0, x_B^1) + 1) \gg 1$, or $\mathrm{lumaAvg} = (\mathrm{Min}(x_A^0, x_A^1) + \mathrm{Max}(x_B^0, x_B^1) + 1) \gg 1$). Similarly, average value of neighbouring chroma samples (chromaAvg) can be calculated by all selected chroma samples, the chroma DC mode value the current chroma CB, or the average of the maximum and minimum chroma samples (e.g., $\mathrm{chromaAvg} = (\mathrm{Max}(y_A^0, y_A^1) + \mathrm{Min}(y_B^0, y_B^1) + 1) \gg 1$, or $\mathrm{chromaAvg} = (\mathrm{Min}(y_A^0, y_A^1) + \mathrm{Max}(y_B^0, y_B^1) + 1) \gg 1$).

[0245] In another sub-embodiment, the default candidates include but not limited to the candidates described below. The default candidates are $\alpha \cdot G + \beta$, where G is the luma sample gradients instead of down-sampled luma samples L. The 16 GLM filters described in the section, entitled: Gradient Linear Model (GLM), are applied. The final scaling parameter α is from the set $\{0, {}^1/_8, -{}^1/_8, +{}^2/_8, -{}^2/_8, +{}^3/_8, -{}^3/_8, +{}^4/_8, -{}^4/_8\}$. The offset parameter $\beta = 1/(1 \ll bit\_depth)$ or is derived based on neighbouring luma and chroma samples.

[0246] In another embodiment, a default candidate can be an earlier candidate with a delta scaling parameter refinement. For example, if the scaling parameter of an earlier candidate is α, the scaling parameter of a default candidate is $(\alpha + \Delta\alpha)$, where $\Delta\alpha$ can be from the set $\{{}^1/_8, -{}^1/_8, +{}^2/_8, -{}^2/_8, +{}^3/_8, -{}^3/_8, +{}^4/_8, -{}^4/_8\}$. The offset parameter of a default candidate will be derived by $(\alpha + \Delta\alpha)$ and the average value of neighbouring luma and chroma samples of the current block.

[0247] In another embodiment, a default candidate can be a shortcut to indicate a cross-component mode (i.e., using the current neighbouring luma/chroma reconstructed samples to derive cross-component models) rather than inheriting parameters from neighbours. For example, a default candidate can be CCLM_LA, CCLM_L, CCLM_A, MMLM_LA, MMLM_L, MMLM_A, single-model CCCM, multiple-model CCCM or cross-component model with a specified GLM pattern.

[0248] In another embodiment, a default candidate can be a cross-component mode (i.e., using the current neighbouring luma/chroma reconstructed samples to derive cross-component models) rather than inheriting parameters from neighbours, and also with a scaling parameter update ($\Delta\alpha$). Then, the scaling parameter of a default candidate is $(\alpha + \Delta\alpha)$. For example, a default candidate can be CCLM_LA, CCLM_L, CCLM_A, MMLM_LA, MMLM_L, or MMLM_A. For another example, $\Delta\alpha$ can be from the set $\{{}^{1}/_{8}, -{}^{1}/_{8}, +{}^{2}/_{8}, -{}^{2}/_{8}, +{}^{3}/_{8}, -{}^{3}/_{8}, +{}^{4}/_{8}, -{}^{4}/_{8}\}$. The offset parameter of a default candidate will be derived by $(\alpha + \Delta\alpha)$ and the average value of neighbouring luma and chroma samples of the current block. For still another example, the $\Delta\alpha$ can be different for each colour components.

[0249] In another embodiment, a default candidate can be an earlier candidate with partial selected model parameters. For example, suppose an earlier candidate has m parameters, it can choose k out of m parameters from an earlier candidate to be a default candidate, where $0 < k < m$ and $m > 1$.

[0250] In another embodiment, a default candidate can be the first model of an earlier MMLM candidate (i.e., the model used when the sample value being less than or equal to a classification threshold). In still another embodiment, a default candidate can be the second model of an earlier MMLM candidate (i.e., the model used when the sample value being greater than or equal to a classification threshold). In still another embodiment, a default candidate can be the combination of two models of an earlier MMLM candidate. For example, if the models of an earlier MMLM candidate are $\{c_0^0, c_1^0, \ldots, c_{M-1}^0\}$ and $\{c_0^1, c_1^1, \ldots, c_{M-1}^1\}$. The model parameters of an default candidate can be $\{(1-\alpha) \times c_0^0 + \alpha \times c_0^1, (1-\alpha) \times c_1^0 + \alpha \times c_1^1, \ldots, (1-\alpha) \times c_{M-1}^0 + \alpha \times c_{M-1}^1\}$, where α is a weighting factor which can be predefined or implicitly derived based on neighbouring template cost, and c_x^y is the x-th parameter of the y-th model.

[0251] In another embodiment, default candidates can be derived from reconstructed samples from non-adjacent neighbouring regions. Let the current block position be at (x, y) and the block size be $w \times h$. If the reconstructed samples in the MxN region located at (x+dx, y+dy) are "available", the default candidates can be derived using reconstructed luma and chroma samples in the region. For example, MxN can be 8x8. For another example, MxN can be 16x8. For another example, MxN can be 16x16. For another example, MxN can be $w \times h$. The meaning of "available" can be the reconstructed sample inside the current block is available, or the reconstructed sample inside the k lines of neighbouring samples is available. The k can be defined by the IBC neighbouring search region, or the neighbouring buffer area of other intra coding tools (e.g., multi-reference line intra prediction, CCLM or CCCM).

[0252] In another embodiment, let the current block position be at (x, y) and the block size be $w \times h$, the default candidates can be derived using reconstructed samples in the MxN region located at (x_{mid} +dx, y_{mid} +dy), if the reconstructed samples in the region are available, where (x_{mid}, y_{mid}) = (x + w/2, y + h/2).

[0253] In another embodiment, default candidates derived from reconstructed samples in non-adjacent neighbouring regions can be any type of cross-component models or some particular types of cross-component models. For example, the derived model can be CCLM, MMLM, CCCM, CCCM multi-models, or other cross-component models. For another example, the derived model is CCCM model. For another example, the derived model is CCLM model. For another example, the derive model is CCCM or CCCM multi-models.

[0254] In another embodiment, assume two value sets α_x and α_y are defined as:

$$\alpha_x = \{\alpha_{x1}, \alpha_{x2}, \alpha_{x3}, \dots, \alpha_{xn}\}, \alpha_{xi} < \alpha_{xj}\ if\ i < j$$

$$\alpha_y = \{\alpha_{y1}, \alpha_{y2}, \alpha_{y3}, \dots, \alpha_{yn}\}, \alpha_{yi} < \alpha_{yj}\ if\ i < j.$$

[0255] All values in α_x and α_y are positive numbers. (dx, dy) can be $(\alpha_{xi} \times w,\ -\alpha_{yi} \times h), (-\alpha_{xi} \times w,\ \alpha_{yi} \times h), (-\alpha_{xi} \times w,\ -\alpha_{yi} \times h), (\alpha_{xi} \times w, 0), (-\alpha_{xi} \times w, 0), (0,\ \alpha_{yi} \times h), (0, y_{mid} - \alpha_{yi} \times h)$.

[0256] In another embodiment, the current block position is at (x, y) and the block size is $w \times h$. Let δx and δy be two fixed positive numbers (dx, dy) can be $(\alpha_{xi} \times \delta x, -\alpha_{yi} \times \delta y), (-\alpha_{xi} \times \delta x, +\alpha_{yi} \times \delta y), (-\alpha_{xi} \times \delta x,\ -\alpha_{yi} \times \delta y), (\alpha_{xi} \times \delta x, 0), (-\alpha_{xi} \times \delta x, 0), (0, \alpha_{yi} \times \delta y)$, or $(0, -\alpha_{yi} \times \delta y)$.

[0257] When constructing a candidate list, candidates are inserted into the list according to a pre-defined order. For example, the pre-defined order can be spatial adjacent candidates, temporal candidates, spatial non-adjacent candidates, historical candidates, and then default candidates. In one embodiment, if cross-component models are derived for non-LM coded blocks (e.g., as mentioned in the section entitled: Inherit Neighbouring Model Parameters for Refining the Cross-Component Model Parameters), the candidate models of non-LM coded blocks are included into the list after including candidate models of LM coded blocks. In another embodiment, if cross-component models are derived for non-LM coded blocks, the candidate models of non-LM coded blocks are included into the list before including default candidates. In still another embodiment, if cross-component models are derived for non-LM coded blocks, the candidate models of non-LM coded blocks have lower priority to be included into the list than candidate models from LM coded blocks.

[0258] When constructing a candidate list, only the candidates with a certain prediction mode can be added into the list. For example, it can limit only the candidates derived by CCLM or MMLM modes being allowed to be added into the list. For another example, it can limit only the candidates derived by single-model modes (e.g., CCLM, or CCLM with single-model) being allowed to be added into the list. For another example, it can limit only the candidates derived by multi-model modes (e.g., MMLM, or CCCM with multi-model) being allowed to be added into the list. For another example, it can limit only the candidates derived by GLM modes being allowed to be added into the list. For another example, it can limit only the candidates derived by a specific mode (e.g., CCLM, MMLM, CCCM, CCCM with multi-model, or GLM) being allowed to be added into the list. In one embodiment, if only the candidate with a certain prediction mode can be added into the list, when doing prediction mode signalling, it can signal the prediction mode first and then signal if the proposed cross-component merge mode is used or not. If the proposed cross-component merge mode is used, then signal the candidate index.

[0259] In the chroma intra fusion mode, a non-CCLM coded intra prediction and a CCLM coded intra prediction are fused together to obtain the final intra prediction. In one embodiment, when inheriting the cross-component model parameters from the block/position coded by chroma intra fusion mode, the model parameters for obtaining the CCLM coded intra prediction are inherited and further refined. In another embodiment, the fusion weight, the coding mode of non-CCLM coded intra prediction and the model parameters for obtaining the CCLM coded intra prediction are inherited and further refined. In still another embodiment, the coding mode of non-CCLM coded intra prediction is implicitly derived (e.g., derived as DM or planar mode), and the fusion weight and the model parameters for obtaining the CCLM coded intra prediction are inherited and further refined. In still another embodiment, if the non-CCLM coded intra prediction of the block/position coded by chroma intra fusion mode can be implicitly derived (e.g., the non-CCLM coded intra prediction is DM or planar mode), the fusion weight and the model parameters for obtaining the CCLM coded intra prediction are inherited and further refined.

[0260] **Removing or Modifying Similar Neighbouring Model Parameters**

[0261] When inheriting cross-component model parameters from other blocks, it can further check the similarity between the inherited model and the existing models in the candidate list or those model candidates derived by the neighbouring reconstructed samples of the current block (e.g., models derived by CCLM, MMLM, or CCCM using the neighbouring reconstructed samples of the current block). If the model of a candidate parameter is similar to the existing models, the model will not be included in the candidate list. In one embodiment, it can compare the similarity of $(\alpha \times \text{lumaAvg} + \beta)$ or α among existing candidates to decide whether to include the model of a candidate or not. For example, if the $(\alpha \times \text{lumaAvg} + \beta)$ or α of the candidate is the same as one of the existing candidates, the model of the candidate is not included. For another example, if the difference of $(\alpha \times \text{lumaAvg} + \beta)$ or α between the candidate and one of existing candidates is less than a threshold, the model of the candidate is not included. Besides, the threshold can be adaptive based on coding information (e.g., the current block size or area). For another example, when comparing the similarity, if a model from a candidate and the existing model both use CCCM, it can compare similarity by checking the value of ($c_0C + c_1N + c_2S + c_3E + c_4W + c_5P + c_6B$) to decide whether to include the model of a candidate or not. In another embodiment, if a candidate position point to a CU which is the same one of the existing candidates, the model of the candidate parameter is not included. In still another embodiment, if the model of a candidate is similar to one of existing candidate models, it can adjust the inherited model parameters so that the inherited model is different from the existing candidate models. For example, if the inherited scaling parameter is similar to one of existing candidate models, the inherited scaling parameter can add a predefined offset (e.g., $1 \gg S$ or $-(1 \gg S)$, where S is the shift parameter) so that the inherited parameter is different from the existing candidate models.

[0262] In another embodiment, only partial model parameters with the existing models in the candidate list are compared. For example, a CCLM candidate has scale and offset parameters, it may only compare to determine if the scale or offset parameters are the same or similar to existing candidates. If the scale or offset parameters are the same or similar, the model will not be included in the candidate list. For another example, a CCCM candidate has c_0 to c_6 parameters, it my only compare if n parameters ($n < 7$) are the same or similar to existing candidates or not. If the scale or offset parameters are the same or similar, the model will not be included in the candidate list.

[0263] In another embodiment, it can apply a candidate model to the neighbouring reconstructed samples of the current block, and compare the difference with the existing candidate models. If the difference value is less than or equal to a threshold, the model will not be included in the candidate list. For example, assume the applied result is p_j^{nei} and the corresponding results of the existing models in the candidate list are p_0^{nei} to p_i^{nei}. If $|p_j^{nei} - p_0^{nei}| < th$, $|p_j^{nei} - p_1^{nei}| < th$, …, or $|p_j^{nei} - p_i^{nei}| < th$, the model will not be included in the candidate list. For the selection of the neighbouring reconstructed samples, it may choose the neighbouring reconstructed sample with the maximal value, the neighbouring reconstructed sample with the minimal value, the mean/median/mode of the neighbouring reconstructed samples, the left-side neighbouring reconstructed samples, the above-side neighbouring reconstructed samples, or the above-left neighbouring reconstructed samples.

[0264] In another embodiment, the number of candidates with the same type (e.g., MMLM, CCCM, or GLM) is limited when including the candidates into the list. For example, if the current list has k candidates with MMLM type, it is not allowed to further include candidates with MMLM type into the list. For another example, if the current list has k candidates with CCCM type, it is not allowed to further include candidates with CCCM type into the list. For another example, if the current list has k candidates with GLM type, it is not allowed to further include candidates with GLM type into the list.

[0265] In another embodiment, default candidates will not be compared with the existing models in the candidate list and will be included in the candidate list.

[0266] **Reordering the Candidates in the List**

[0267] The candidates in the list can be reordered to reduce the syntax overhead when signalling the selected candidate index. The reordering rules can depend on the coding information of neighbouring blocks or the model error. For example, if neighbouring above or left blocks are coded by MMLM, the MMLM candidates in the list can be moved to the head of the current list. Similarly, if neighbouring above or left blocks are coded by single model LM or CCCM, the single model LM or CCCM candidates in the list can be moved to the head of the current list. Similarly, if GLM is used by neighbouring above or left blocks, the GLM related candidates in the list can be moved to the head of the current list.

[0268] In still another embodiment, the reordering rule is based on the model error by applying the candidate model to the neighbouring templates of the current block, and then compare the error with the reconstructed samples of the neighbouring template. For example, as shown in Fig. 27, the size of above neighbouring template 2720 of the current block is $w_a \times h_a$, and the size of left neighbouring template 2730 of the current block 2710 is $w_b \times h_b$. Suppose K models are in the current candidate list, and α_k and β_k are the final scale and offset parameters after inheriting the candidate k. The model error of candidate k corresponding to the above neighbouring template is:

$$e_a^k = \sum_{i,j} \left| \left(\alpha_k \times recL'^{(i,j)}_a + \beta_k \right) - recC_a^{(i,j)} \right|$$

[0269] where, $recL'^{(i,j)}_a$ and $recC_a^{(i,j)}$ are the reconstructed samples of luma (e.g., after downsampling process or after applying GLM pattern) and reconstructed samples of chroma at position (i,j) in the above template, and $0 \le i < w_a$ and $0 \le j < h_a$.

[0270] Similarly, the model error of candidate k by the left neighbouring template is:

$$e_b^k = \sum_{m,n} \left| \left(\alpha_k \times recL'^{(m,n)}_b + \beta_k \right) - recC_b^{(m,n)} \right|$$

[0271] where $recL'^{(m,n)}_b$ and $recC_b^{(m,n)}$ are the reconstructed samples of luma (e.g., after applying downsampling process or GLM pattern) and reconstructed samples of chroma at position (m, n) in the left template, and $0 \le m < w_b$ and $0 \le n < h_b$.

[0272] Then the model error of candidate k is:

$$e^k = e_a^k + e_b^k$$

[0273] After calculating the model error among all candidates, it can get a model error list $E = \{e^0, e^1, e^2, \dots, e^k, \dots, e^K\}$. Then, it can reorder the candidate index in the inherited candidate list by sorting the model error list in ascending order.

[0274] In still another embodiment, if the candidate k uses CCCM prediction, the e_a^k and e_b^k are defined as:

$$e_a^k = \sum_{i,j} \left| \left(c0_k \times recL'^{(i,j)}_a + c1_k \times recL'^{(i,j-1)}_a + c2_k \times recL'^{(i,j+1)}_a + c3_k \times recL'^{(i-1,j)}_a + c4_k \times recL'^{(i+1,j)}_a + c5_k \times P + c6_k \times B \right) - recC^{(i,j)}_a \right|$$

$$e_b^k = \sum_{i,j} \left| \left(c0_k \times recL'^{(i,j)}_b + c1_k \times recL'^{(i,j-1)}_b + c2_k \times recL'^{(i,j+1)}_b + c3_k \times recL'^{(i-1,j)}_b + c4_k \times recL'^{(i+1,j)}_b + c5_k \times P + c6_k \times B \right) - recC^{(i,j)}_b \right|$$

where $c0_k, c1_k, c2_k, c3_k, c4_k, c5_k$, and $c6_k$ are the final filtering coefficients after inheriting the candidate k. P and B are the nonlinear term and bias term.

[0275] In still another embodiment, if the above neighbouring template is not available, then $e^k = e_b^k$. Similarly, if the left neighbouring template is not available, then $e^k = e_a^k$. If both templates are not available, the candidate index reordering method using model error is not applied.

[0276] In still another embodiment, not all positions inside the above and left neighbouring template are used in calculating model error. It can choose partial positions inside the above and left neighbouring template to calculate model error. For example, it can define a first start position and a first subsampling interval depends on the width of the current block to partially select positions inside the above neighbouring template. Similarly, it can define a second start position and a second subsampling interval depends on the height of the current block to partially select positions inside the left neighbouring template. For another example, h_a or w_b can be a constant value (e.g., h_a or w_b can be 1, 2, 3, 4, 5, or 6). For another example, h_a or w_b can be dependent on the block size. If the current block size is greater than or equal to a threshold, h_a or w_b is equal to a first value. Otherwise, h_a or w_b is equal to a second value.

[0277] In still another embodiment, the candidates of different types are reordered separately before the candidates are added into the final candidate list. For each type of the candidates, the candidates are added into a primary candidate list with a pre-defined size N_1. The candidates in the primary list are reordered. The candidates (N_2) with the smallest costs are then added into the final candidate list, where $N_2 \leq N_1$. In another embodiment, the candidates are categorized into different types based on the source of the candidates, including but not limited to the spatial neighbouring models, temporal neighbouring models, non-adjacent spatial neighbouring models, and the historical candidates. In another embodiment, the candidates are categorized into different types based on the cross-component model mode. For example, the types can be CCLM, MMLM, CCCM, and CCCM multi-model. For another example, the types can be GLM-non active or GLM active.

[0278] In still another embodiment, after the candidates are reordered based on the template cost, the redundancy of the candidate can be further checked. A candidate is considered to be redundant if the template cost difference between it and its predecessor in the list is smaller than a threshold. If a candidate is considered redundant, it can be removed from the list, or it can be move to the end of the list.

[0279] **Inheriting Candidates from the Candidates in the Candidate List of Neighbours**

[0280] The candidates in the current inherited candidate list can be from neighbouring blocks. For example, it can inherit the first *k* candidates in the inherited candidate list of the neighbouring blocks. As shown in the Fig. 28, the current block can inherit the first two candidates in the inherited candidate list of the above neighbouring block and the first two candidates in the inherited candidate list of the left neighbouring block. For an embodiment, after adding the neighbouring spatial candidates and non-adjacent spatial candidates, if the current inherited candidate list is not full, the candidates in the candidate list of neighbouring blocks are included into the current inherited candidate list. For another embodiment, when including the candidates in the candidate list of neighbouring blocks, the candidates in the candidate list of left neighbouring blocks are included before the candidates in the candidate list of above neighbouring blocks. For still another embodiment, when including the candidates in the candidate list of neighbouring blocks, the candidates in the candidate list of above neighbouring blocks are included before the candidates in the candidate list of left neighbouring blocks.

[0281] **Signalling the Inherit Candidate Index in the List**

[0282] An on/off flag can be signalled to indicate if the current block inherits the cross-component model parameters from neighbouring blocks or not. The flag can be signalled per CU/CB, per PU, per TU/TB, or per colour component, or per chroma colour component. A high level syntax can be signalled in SPS, PPS (Picture Parameter Set), PH (Picture header) or SH (Slice Header) to indicate if the proposed method is allowed for the current sequence, picture, or slice.

[0283] The maximum allowed candidate number is signalled to indicate the maximum size of the merge candidate list. The number can be signalled per CU/CB, per PU, per TU/TB, or per colour component, or per chroma colour component. A high level syntax can be signalled in SPS, PPS, PH or SH to indicate if the proposed method is allowed for the current sequence, picture, or slice. The maximum allowed candidate number of the proposed method (i.e., CCP merge mode) can be shared with the maximum allowed candidate number for inter merge mode.

[0284] If the current block inherits the cross-component model parameters from neighbouring blocks, the inherited candidate index is signalled. The index can be signalled (e.g., signalled using truncate unary code, Exp-Golomb code, or fix length code) and shared between the current Cb and Cr blocks. For another example, the index can be signalled per colour component. For example, one inherited index is signalled for Cb component, and another inherited index is signalled for Cr component. For another example, it can use chroma intra prediction syntax (e.g., IntraPredModeC[xCb][yCb]) to store the inherited index.

[0285] If the current block inherits the cross-component model parameters from neighbouring blocks, the current chroma intra prediction mode (e.g., IntraPredModeC[xCb][yCb] as defined in VVC standard) is temporally set to a cross-component mode (e.g., CCLM_LA) at the bitstream syntax parsing stage. Later, at the prediction stage or reconstruction stage, the candidate list is derived, and the inherited candidate model is then determined by the inherited candidate index. After obtaining the inherited model, the coding information of the current block is then updated according to the inherited candidate model. The coding information of the current block includes but not limited to the prediction mode (e.g., CCLM_LA or MMLM_LA), related sub-mode flags (e.g., CCCM mode flag), prediction pattern (e.g., GLM pattern index), and the current model parameters. Then, the prediction of the current block is generated according to the updated coding information.

[0286] **Removing or Modifying Similar Model Parameters when Adding Candidates into a History Table**

[0287] When adding a cross-component model into a history table, it may further check the similarity between the to-be-added model and the existing models in the history table. If the to-be-added model is similar to the existing models, the to-be-added model will not be included in the history table. In one embodiment, it may compare the similarity of $(\alpha \times \text{lumaAvg} + \beta)$ or α with existing candidates to decide whether to include the to-be-added model or not. For example, if the $(\alpha \times \text{lumaAvg} + \beta)$ or α of the to-be-added model is the same as one of the existing candidates, the to-be-added model is not included. For another example, if the difference of $(\alpha \times \text{lumaAvg} + \beta)$ or α between the to-be-added model and one of existing models is less than a threshold, the to-be-added model is not included. Besides, the threshold can be adaptive based on coding information (e.g., the current block size or area). For another example, when comparing the similarity, if both a to-be-added model and the existing model use CCCM, it may compare similarity by checking the value of $(c_0C + c_1N + c_2S + c_3E + c_4W + c_5P + c_6B)$ to decide whether to include the to-be-added model or not. In another embodiment, if the CU position of the current to-be-added model is the same CU position of the existing candidates, the to-be-added model parameter is not included. In still another embodiment, if the to-be-added model is similar to one of existing candidate models, it may adjust the inherited model parameters to let the to-be-added model be different from the existing candidate models. For example, if the to-be-added scaling parameter is similar to one of existing candidate models, the to-be-added scaling parameter may add a predefined offset (e.g., $1 \gg S$ or $-(1 \gg S)$, where S is the shift parameter) to let the to-be-added model be different from the existing candidate models.

[0288] In another embodiment, only partial model parameters are compared with the existing models in the history table. For example, a CCLM candidate has scale and offset parameters and it may only compare to determine if the scale or offset parameters are the same or similar to existing candidates. If the scale or offset parameters is the same or similar, the to-be-added model would not be included into the history table. For another example, a CCCM candidate has c_0 to c_6 parameters, it could only compare if n parameters ($n < 7$) are the same or similar with existing candidates or not. If the scale or offset parameters are the same or similar, the to-be-added model will not be included in the history table.

[0289] In another embodiment, it may apply a to-be-added model to the neighbouring reconstructed samples of the current block, and compare the difference with the existing candidate models. If the difference value is less than or equal to a threshold, the to-be-added model will not be included in the history table. For example, assume the applied result is p_j^{nei} and the corresponding results of the existing models in the history table are p_0^{nei} to p_i^{nei}. If $|p_j^{nei} - p_0^{nei}| < th$, $|p_j^{nei} - p_1^{nei}| < th$, ..., or $|p_j^{nei} - p_i^{nei}| < th$, the to-be-added model will not be included into the history table. For the selection of the neighbouring reconstructed samples, it may choose the neighbouring reconstructed sample with the maximal value, the neighbouring reconstructed sample with the minimal value, the mean/median/mode of the neighbouring reconstructed samples, the left-side neighbouring reconstructed samples, the above-side neighbouring reconstructed samples, or the above-left neighbouring reconstructed samples.

[0290] In another embodiment, the number of candidates with the same type (e.g., MMLM, CCCM, or GLM) is limited when including the candidates into the history table. For example, if the current history table has k candidates with MMLM type, it is not allowed to further include candidates with MMLM type into the history table. For another example, if the current history table has k candidates with CCCM type, it is not allowed to further include candidates with CCCM type into the history table. For another example, if the current history table has k candidates with GLM type, it is not allowed to further include candidates with GLM type into the history table.

[0291] In another embodiment, the constraints, or rules to prevent adding a redundant candidate into a history table will share/be the same as that of preventing adding a redundant candidate into a candidate list (e.g., the constraints or rules mentioned in the section entitled: Removing or Modifying Similar Neighbouring Model Parameters)).

[0292] **Inheriting Multiple Cross-Component Models**

[0293] The final prediction of the current block can be the combination of multiple cross-component models, or fusion of the selected cross-component models with the prediction by non-cross-component coding tools (e.g., intra angular prediction modes, intra planar/DC modes, or inter prediction modes). In one embodiment, if the current candidate list size is N, it can select k candidates from the total N candidates (where $k \leq N$). Then, k predictions are respectively generated by applying the cross-component model of the selected k candidates using the corresponding luma reconstructed samples. The final prediction of the current block is the combination results of these k predictions. For example, if two candidate predictions (denoted as p_{cand1} and p_{cand2}) are combined, the final prediction at (x, y) position of the current block is $p_{final}(x,y) = (1-\alpha) \times p_{cand1}(x,y) + \alpha \times p_{cand2}(x,y)$, where α is a weighting factor. Besides, the weighting factor α can be predefined or implicitly derived by neighbouring template cost. For example, by using the template cost defined in the section entitled: Inherit Non-Adjacent Spatial Neighbouring Models, the corresponding template cost of two candidates are e^{cand1} and e^{cand2}, then α is $e^{cand1}/(e^{cand1} + e^{cand2})$. In another embodiment, if two candidate models are combined, the selected models are from the first two candidates in the list. In still another embodiment, if i candidate models are combined, the selected models are from the first i candidates in the list.

[0294] In another embodiment, if the current candidate list size is N, it can select k candidates from the total N candidates (where $k \leq N$). The k cross-component models can be combined into one final cross-component model by weighted-averaging the corresponding model parameters. For example, if a cross-component model has M parameters, the j-th parameter of the final cross-component model is the weighted-averaging of the j-th parameter of the k selected candidate, where j is 1 … M. Then, the final prediction is generated by applying the final cross-component model to the corresponding luma reconstructed samples. For example, if two candidate models are $\{c_0^0, c_1^0, \ldots, c_{M-1}^0\}$ and $\{c_0^1, c_1^1, \ldots, c_{M-1}^1\}$. The final cross-component model is $\{(1-\alpha) \times c_0^0 + \alpha \times c_0^1, (1-\alpha) \times c_1^0 + \alpha \times c_1^1, \ldots, (1-\alpha) \times c_{M-1}^0 + \alpha \times c_{M-1}^1\}$, where α is a weighting factor which can be predefined or implicitly derived by neighbouring template cost, and c_x^y is the x-th model parameter of the y-th candidate. For example, by using the template cost defined in the section entitled: Inherit Non-Adjacent Spatial Neighbouring Models, the corresponding template cost of two candidates are e^{cand1} and e^{cand2}, then α is $e^{cand1}/(e^{cand1} + e^{cand2})$. For still an example, the two candidate models are one from the spatial adjacent neighbouring candidate, and another one from the non-adjacent spatial candidate or history candidate. If the spatial adjacent neighbouring candidate is not available, then the two candidate models are all from the non-adjacent spatial candidates or history candidates. In another embodiment, if two candidate models are combined, the selected models are from the first two candidates in the list. In still another embodiment, if i candidate model is combined, the selected models are from the first i candidate in the list.

[0295] In another embodiment, two cross-component models are combined into one final model by weighted-averaging the corresponding model parameters, where the two cross-component models are one from the above spatial neighbouring candidate and another one from the left spatial neighbouring candidate. The above spatial neighbouring candidate is the neighbouring candidate that has the vertical position less than or equal to the top block boundary position of the current block. The left spatial neighbouring candidate is the neighbouring candidate that has the horizontal position less than or equal to the left block boundary position of the current block. The weighting factor α is determined according to the horizontal and vertical spatial positions inside the current block. For example, if two candidate predictions (denoted as p_{above} and p_{left}) are combined, the final prediction at (x, y) position of the current block is $p_{final}(x,y) = (1-\alpha) \times p_{above}(x,y) + \alpha \times p_{left}(x,y)$, where $\alpha = y/(x+y)$. In another embodiment, the above spatial neighbouring candidate is the first candidate in the list that has the vertical position less than or equal to the top block boundary position of the current block. The left spatial neighbouring candidate is the first candidate in the list that has the horizontal position less than or equal to the left block boundary position of the current block.

[0296] In another embodiment, it can combine cross-component model candidates with the prediction by non-cross-component coding tools. For example, one cross-component model candidate is selected from a list, and its prediction is denoted as p_{ccm}. Another prediction can be from chroma DM, chroma DIMD, or intra angular mode, and denoted as $p_{curr-ccm}$. The final prediction at (x, y) position of the current block is $p_{final}(x,y) = (1-\alpha) \times p_{ccm}(x,y) + \alpha \times p_{curr-ccm}(x,y)$, where α is the weighting factor, which can be predefined or implicitly derived by neighbouring template cost. For still the same example, the prediction by a non-cross-component coding tool can be predefined or signalled. The prediction by non-cross-component coding tool is chroma DM or chroma DIMD. For another example, prediction by non-cross-component coding tool is signalled, but the index of cross-component model candidate is predefined or determined by the coding modes of neighbouring blocks. For still the same example, if at least one of neighbouring spatial blocks is coded with CCCM mode, the first candidate has CCCM model parameters is selected. If at least one of neighbouring spatial blocks is coded with GLM mode, the first candidate has GLM pattern parameters is selected. Similarly, if at least one of neighbouring spatial blocks is coded with MMLM mode, the first candidate has MMLM parameters is selected.

[0297] In another embodiment, it can combine cross-component model candidates with the prediction by the current cross-component model. For example, one cross-component model candidate is selected from the list, and its prediction is denoted as p_{ccm}. Another prediction can be from the cross-component prediction mode by the current neighbouring reconstructed samples and denoted as $p_{curr-ccm}$. The final prediction at (x, y) position of the current block is $p_{final}(x,y) = (1-\alpha) \times p_{ccm}(x,y) + \alpha \times p_{curr-ccm}(x,y)$, where α is the weighting factor which can be predefined or implicitly derived according to neighbouring template cost. For still the same example, the prediction by the current cross-component model can be predefined or signalled. The prediction by non-cross-component coding tool is chroma DM or chroma DIMD. For another example, prediction by non-cross-component coding tool is signalled, but the index of cross-component model candidate is predefined or determined by neighbouring blocks coding mode. For still the same example, if at least one of neighbouring spatial blocks is coded with CCCM mode, the first candidate has CCCM model parameters is selected. If at least one of neighbouring spatial blocks is coded with GLM mode, the first candidate has GLM pattern parameters is selected. Similarly, if at least one of neighbouring spatial blocks is coded with MMLM mode, the first candidate has MMLM parameters is selected.

[0298] In another embodiment, it can combine multiple cross-component models into one final cross-component model. For example, it can choose one model from a candidate, and choose a second model from another candidate to be a multi-model mode. The selected candidate can be CCLM/MMLM/GLM/CCCM coded candidate. The multi-model classification threshold can be the average of the offset parameters (e.g., offset/β in CCLM, or $c_6 \times B$ or c_6 in CCCM) of the two selected modes. In one embodiment, if two candidate models are combined, the selected models are the first two candidates in the list. In another embodiment, the classification threshold is set to the average value of the neighbouring luma and chroma samples of the current block.

[0299] **Refining the Inherited Candidate Positions**

[0300] In one embodiment, the final inherited model of the current block is from the cross-component model at the indicated candidate position with a delta position. For example, if the current selected candidate position is (x_{nei}^i, y_{nei}^i), it can further signal a delta position, (dx_{nei}^i, dy_{nei}^i), to indicate the position of the final inherited model. That is, the final inherited model of the current block is from the cross-component model at $(x_{nei}^i + dx_{nei}^i, y_{nei}^i + dy_{nei}^i)$. In one embodiment, the signal delta position can only have a horizontal delta position or a vertical delta position, that is, $(dx_{nei}^i, 0)$ or $(0, dy_{nei}^i)$. Besides, the signalled delta position can be shared among multiple colour components or signalled per colour component. For example, the signalled delta position is share for the current Cb and Cr blocks, or the signalled delta position is only used for the current Cb block or the current Cr block. Furthermore, the signalled dx_{nei}^i or dy_{nei}^i may have a sign bit to indicate positive delta position or negative delta position. When indicating the magnitude of dx_{nei}^i or dy_{nei}^i, it can be signalled by a look-up table index. For example, a look-up table is {1, 2, 4, 8, 16, …}, if $|dx_{nei}^i|$ is equal to 8, then the table index 3 is signalled (the first table index is 0).

[0301] In one embodiment, when a candidate is selected from the candidate list, the models from the neighbouring positions of the selected candidate are further searched. The final inherited model can be from the neighbouring position of the selected candidate. Positions of a pre-defined search pattern inside an area around the selected candidate is searched. In one embodiment, the neighbouring positions searched are either horizontally different or vertically different from the selected candidate, that is, the delta position is either $(dx_{nei}^i, 0)$ or $(0, dy_{nei}^i)$. In another embodiment, the neighbouring positions searched are diagonally different from the selected candidate, that is, the delta position is (dx_{nei}^i, dy_{nei}^i), where $dx_{nei}^i = \pm dy_{nei}^i$. Note, the delta position can be a positive or negative number.

[0302] In another embodiment, the models from the neighbouring positions of the candidate are further searched only when the selected candidate is a non-adjacent candidate. Positions of a pre-defined search pattern inside an area around the selected candidate are searched. For example, suppose the distances between the non-adjacent candidates are the current coding block width and height. After a non-adjacent candidate is selected, the positions whose horizontal distance and vertical distance are both smaller than current coding block width and height respectively are further searched, i.e., dx_{nei}^{i} is within the range of $\pm width$ and dy_{nei}^{i} is within the range of $\pm height$. In one embodiment, the neighbouring positions searched are either horizontally different or vertically different from the selected candidate, that is, the delta position is either $(dx_{nei}^{i}, 0)$ or $(0, dy_{nei}^{i})$. In another embodiment, the neighbouring positions searched are diagonally different from the selected candidate, that is, the delta position is $(dx_{nei}^{i}, dy_{nei}^{i})$, where $dx_{nei}^{i} = \pm dy_{nei}^{i}$.

[0303] **Inheriting from Shared Cross-Component Models**

[0304] In one embodiment, the current picture is segmented into multiple non-overlapped regions, and each region size is $M \times N$. A shared cross-component model is derived for each region, respectively. The neighbouring available luma/chroma reconstructed samples of the current region are used to derive the shared cross-component model of the current region. Then, for a block inside the current region, it can determine whether to inherit the shared cross-component model or derive cross-component model by the neighbouring available luma/chroma reconstructed samples of the block. In one embodiment, the $M \times N$ can be a predefined value (e.g. 32x32 regarding to the chroma format), a signalled value (e.g. signalled in sequence/picture/slice/tile-level), a derived value (e.g. depending on the CTU size), or the maximum allowed transform block size.

[0305] In another embodiment, each region may have more than one shared cross-component model. For example, it can use various neighbouring templates (e.g., top and left neighbouring samples, top-only neighbouring samples, left-only neighbouring samples) to derive more than one shared cross-component model. Besides, the shared cross-component models of the current region can be inherited from previously used cross-component models. For example, the shared model can be inherited from the models of adjacent spatial neighbours, non-adjacent spatial neighbours, temporal neighbours, or from a historical list.

[0306] When doing signalling, a first flag can be used to determine if the current cross-component model is inherited from the shared cross-component models or not. If the current cross-component model is inherited from the shared cross-component models, the second syntax indicate the inherited index of the shared cross-component models (e.g., signalled using truncate unary code, Exp-Golomb code, or fixed length code).

[0307] **Sharing the Buffer Resource with Existing Coding Tools**

[0308] To store the CCM information (e.g., prediction mode, related sub-mode flags, prediction pattern, or model parameters) for further model inheritance, the buffer for storing inter coding information (e.g., motion vector buffer) is shared with cross-component merge mode for storing CCM information. The buffer size can be reduced by sharing the buffer among different coding tools. Otherwise, buffer space has to be allocated to store CCM information and inter coding information separately. The key idea behind the shared buffer is that a block is coded using only one selected coding mode among multiple candidates. Therefore, the coding information for various coding modes can share a common buffer. Suppose the minimal allowed block size is $m \times n$, the current CTU size is $p \times q$, and the current picture size is $r \times s$. A CTU-level buffer and picture-level buffers are used for storing the inter coding and CCM information of the current CTU and each picture, respectively. A CTU-level buffer is created for storing the final inter coding or CCM information, and this CTU-level buffer size is $\lceil p/m \rceil \times \lceil q/n \rceil$. Since $\lceil p/m \rceil$ corresponds to the number of blocks in the horizontal direction and $\lceil q/n \rceil$ corresponds to the number of blocks in the vertical direction, $\lceil p/m \rceil \times \lceil q/n \rceil$ corresponds to the total number of blocks in the CTU. A picture-level buffer is created for storing the final inter coding or CCM information of the current picture, and this picture-level buffer size is $\lceil r/i \rceil \times \lceil s/j \rceil$, where $i \geq m$ and $j \geq n$. In other words, the coding information is stored in the picture buffer in the unit of $i \times j$. Since $\lceil r/i \rceil$ corresponds to the number of second blocks in the horizontal direction and $\lceil s/j \rceil$ corresponds to the number of second blocks in the vertical direction, $\lceil r/i \rceil \times \lceil s/j \rceil$ corresponds to the total number of second blocks in the picture. After encoding or decoding the current block, the inter coding or CCM information of the current block is firstly saved to the corresponding positions of CTU-level buffer in unit of $m \times n$, where the corresponding positions are the positions covered by the current block in unit of $m \times n$. Later, after encoding or decoding the current CTU, the inter coding or CCM information in the current CTU-level buffer is saved to the corresponding positions of the picture-level buffer in unit of $i \times j$.

[0309] However, if the unit of CTU-level buffer and picture-level buffer are not the same (e.g., $i > m$ or $j > n$), it should subsample the inter coding or CCM information in CTU-level buffer for saving to the picture-level buffer. Suppose $i/m = g$ and $j/n = h$, one of each $g \times h$ grids of CTU-level buffer is selected to save the inter coding or CCM information to the corresponding position of the picture-level buffer. For example, as shown in Fig. 29, if $g = 2$ and $h = 2$, one selected position of each 2x2 grid is selected to save the inter coding or CCM information to the corresponding position of the picture-level buffer. In one embodiment, the selected position can be the left-above, left-bottom, right-above, or right-bottom of each 2x2 grid. As shown in Fig. 29, the inter coding or CCM information at the left-above position marked in slash of each 2x2 grid is saved to the picture-level buffer. In another embodiment, when subsampling the CCM information in CTU-level buffer for saving to the picture-level buffer, it can conditionally check the prediction modes inside the $g \times h$ grids. For example, if more than a percentage of positions inside the $g \times h$ grids are intra mode (e.g., more than 50% or 75%), the selected and saved data is CCM information. Otherwise (i.e., most of positions inside the $g \times h$ grids are inter mode), the selected and saved data is inter coding information. When selecting the candidate for saving to picture-level buffer, it can follow a predefined scanning order to select the first allowed candidate. For example, if the selected and saved data is CCM information, it can select the first grid inside the $g \times h$ grids that has CCM information by a predefined scanning order. For another example, if the selected and saved data is inter coding information, it can select the first grid inside the $g \times h$ grids that has inter coding information by a predefined scanning order.

[0310] Due to the buffer for storing inter coding information is shared with cross-component merge mode, it can check the CU prediction mode (e.g., intra prediction, or inter prediction) to identify if the information stored at a certain buffer position is inter coding or CCM information. In one embodiment, if the CU prediction mode is intra prediction, the stored information is CCM information. Otherwise (i.e., CU prediction mode is non-intra prediction), the stored information is inter coding information. In another embodiment, it can set an invalid inter prediction reference index or invalid MV value (e.g., horizontal or vertical MV value) to identify the stored information is CCM information. Otherwise (i.e., valid inter prediction index), the stored information is inter coding information. For example, in VVC standard specification, the inter prediction reference index greater than 2 is invalid, then it may set inter prediction reference index to a value greater than 2 to identify the stored information is CCM information (e.g., inter prediction reference index is 3).

[0311] **Region-wise Cross-Component Model Merge Method**

[0312] According to this method, a current block is partitioned into two or more prediction regions/sub-blocks, where each region can be predicted by inter or intra coding tool. Furthermore, at least one of prediction regions is coded by CC merge mode, where the cross-component model of the at least one region is inherited from a spatial, historical, or temporal neighbouring block/position. In one embodiment, the current block is partitioned by quad-tree, binary-tree, or ternary-tree split. The split can be symmetric or asymmetric split.

[0313] In another embodiment, the current block is partition into two regions, where one of two regions is predicted by inter or intra coding tool, and another one is predicted by CC merge mode. The inherited candidate index of the region predicted by CC merge mode can be explicitly or implicitly indicated. For example, it can explicitly signal the candidate index by the method in section entitled: Signalling the Inherited Candidate Index in the List. For another example, it can implicitly select the first candidate in the list as the candidate index. The candidates in the list can be reordered by the method mentioned in the section entitled: Reordering the Candidates in the List.

[0314] In still another embodiment, current block is partition into two regions, two regions are both predicted by CC merge mode, and the first two candidates in the list are the candidate indexes of the two regions. It can implicitly set the candidate index of the first region (e.g., the region having the top-left sample of the current block) to the first candidate and set the candidate index of the second region to the second candidate. Besides, the list can be reordered by the method mentioned in the section entitled: Reordering the Candidates in the List. For another example, if two regions are both predicted by CC merge mode, an index is explicitly signalled to indicate the candidate index of the first region, and the candidate index of the second region is the signalled index + k or the signalled index – k, where k can be 1, 2, 3, 4, or 5. For still the same example, the candidate index of the first region is implicitly derived from the stored cross-component model at the top-left position of the current block relative to the top-left position in the previous coded slices/pictures as the method mentioned in the section entitled: Inheriting Temporal Neighbouring Model Parameters. In another embodiment, if two regions are both predicted by CC merge mode, the first candidate in the list is the candidate index of the two regions.

[0315] **Cross-Component Skip Mode**

[0316] With the help of many cross-component models, intra chroma prediction becomes more accurate, and the entropy of residue is also getting smaller. Also, the probability that all the CBFs (Coding Block Flags) or coded flags of chroma components in the intra coding mode are equal to 0 has increased compared with the previous video coding standards. Based on this observation, several methods related to a skip mode shortcut are proposed to improve the coding performance of intra chroma prediction.

[0317] In one embodiment, a skip flag can be signalled before the merge flag which indicates whether the current coded block inherits the cross-component model parameters from the neighbouring or not.

[0318] In one embodiment, if the skip flag is true (i.e., the skip mode being used), the merge flag will be inferred to 1, the index of inherited candidate will be signalled directly, and the CBF of the Cb component and the CBF of the Cr component are both inferred to 0 as shown in Fig. 30. In Fig. 30, when the skip flag (i.e. skipFlag) is true (i.e., having a value of 1), the index of inherited candidate (i.e., mergeIdx) is signalled (3010). The merge flag inferred to 1 and the CBFs of the Cb and Cr components both inferred to 0 are indicated by the dash-lined box 3012.

[0319] In another embodiment, if the skip flag is true, the merge flag will be inferred to 0, the CCP mode index will be signalled, and the CBF of the Cb component and the CBF of the Cr component are both inferred to 0 as shown in Fig. 31. In Fig. 31, when the skip flag (i.e. skipFlag) is true (i.e., having a value of 1), the index of inherited candidate (i.e., LMModeIdx) is signalled (3110). The merge flag inferred to 0 and the CBFs of the Cb and Cr components both inferred to 0 are indicated by the dash-lined box 3112.

[0320] In one embodiment, a high-level syntax can be signalled in SPS, PPS, PH or SH to indicate if the proposed skip flag is allowed for the current sequence, picture, or slice.

[0321] The improved syntax coding for cross-component prediction described above can be implemented in an encoder side or a decoder side. For example, any of the proposed syntax coding method for cross-component prediction can be implemented in an Intra/Inter coding module (e.g. Intra Pred. 150/MC 152 in Fig. 1B) in a decoder or an Intra/Inter coding module is an encoder (e.g. Intra Pred. 110/Inter Pred. 112 in Fig. 1A). Any of the proposed shared buffer to store coding information among multiple coding tools including the CCM mode can also be implemented as a circuit coupled to the intra/inter coding module at the decoder or the encoder. However, the decoder or encoder may also use additional processing unit to implement the required cross-component prediction processing. While the Intra Pred. units (e.g. unit 110/112 in Fig. 1A and unit 150/152 in Fig. 1B) are shown as individual processing units, they may correspond to executable software or firmware codes stored on a media, such as hard disk or flash memory, for a CPU (Central Processing Unit) or programmable devices (e.g. DSP (Digital Signal Processor) or FPGA (Field Programmable Gate Array)).

[0322] Fig. 32 illustrates a flowchart of an exemplary video decoding system that decodes syntax data by using one or more context models according to information associated with the cross-component prediction according to an embodiment of the present invention. The steps shown in the flowchart may be implemented as program codes executable on one or more processors (e.g., one or more CPUs) at the encoder side. The steps shown in the flowchart may also be implemented based hardware such as one or more electronic devices or processors arranged to perform the steps in the flowchart. According to the method for the decoder side, coded data associated with a current block comprising a first-colour block and a second-colour block are received in step 3210, wherein the current block is coded using CCP (Cross-Component Prediction). Coded syntax data corresponding to one or more syntaxes related to transform coding or residual coding applied to the current block are parsed in step 3220. The coded syntax data are decoded to generate said one or more syntaxes by using one or more context models according to information associated with the CCP in step 3230. The current block is decoded using the CCP in step 3240, wherein said decoding comprises the transform coding of the current block or the residual coding of the current block using said one or more syntaxes.

[0323] Fig. 33 illustrates a flowchart of an exemplary video encoding system that encodes syntax data by using one or more context models according to information associated with the cross-component prediction according to an embodiment of the present invention. According to the method for the decoder side, input data associated with a current block comprising a first-colour block and a second-colour block are received in step 3310. The current block is encoded by using CCP (Cross-Component Prediction) in step 3320, wherein encoding process of the current block comprises transform coding to generate transformed data of the current block or residual coding to generate residual data of the current block. One or more syntaxes related to the transform coding or the residual coding are encoded to generate coded syntax data by using one or more context models according to information associated with the CCP in step 3330. The coded syntax data is signalled in step 3340.

[0324] Fig. 34 illustrates a flowchart of an exemplary video coding system that encodes or decoded a current block uses information comprising a chroma syntax corresponding to a coded block flag shared by said two or more chroma blocks according to an embodiment of the present invention. Input data associated with a current block comprising a luma block and two or more chroma blocks are received in step 3410, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction). The current block is encoded or decoded by using the CCP in step 3420, wherein said encoding or decoding uses information comprising a chroma syntax corresponding to one or more coded block flags comprising a root chroma coded block flag.

[0325] Fig. 35 illustrates a flowchart of an exemplary video coding system that uses a merge flag to indicate whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not according to an embodiment of the present invention. According to this method, input data associated with a current block comprising a luma block and two or more chroma blocks are received in step 3510, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction). A merge candidate list is generated for the current block in step 3520, wherein the merge candidate list includes an inherited CCP candidate from a neighbouring block. A merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not is signalled or parsed in step 3530. A skip flag is signalled or parsed before the merge flag in step 3540, wherein when the skip flag is true, the merge flag is inferred. The current block is encoded or decoded by using information comprising the merge candidate list in step 3550.

[0326] The flowcharts shown are intended to illustrate an example of video coding according to the present invention. A person skilled in the art may modify each step, re-arranges the steps, split a step, or combine steps to practice the present invention without departing from the spirit of the present invention. In the disclosure, specific syntax and semantics have been used to illustrate examples to implement embodiments of the present invention. A skilled person may practice the present invention by substituting the syntax and semantics with equivalent syntax and semantics without departing from the spirit of the present invention.

[0327] The above description is presented to enable a person of ordinary skill in the art to practice the present invention as provided in the context of a particular application and its requirement. Various modifications to the described embodiments will be apparent to those with skill in the art, and the general principles defined herein may be applied to other embodiments. Therefore, the present invention is not intended to be limited to the particular embodiments shown and described, but is to be accorded the widest scope consistent with the principles and novel features herein disclosed. In the above detailed description, various specific details are illustrated in order to provide a thorough understanding of the present invention. Nevertheless, it will be understood by those skilled in the art that the present invention may be practiced.

[0328] Embodiment of the present invention as described above may be implemented in various hardware, software codes, or a combination of both. For example, an embodiment of the present invention can be one or more circuit circuits integrated into a video compression chip or program code integrated into video compression software to perform the processing described herein. An embodiment of the present invention may also be program code to be executed on a Digital Signal Processor (DSP) to perform the processing described herein. The invention may also involve a number of functions to be performed by a computer processor, a digital signal processor, a microprocessor, or field programmable gate array (FPGA). These processors can be configured to perform particular tasks according to the invention, by executing machine-readable software code or firmware code that defines the particular methods embodied by the invention. The software code or firmware code may be developed in different programming languages and different formats or styles. The software code may also be compiled for different target platforms. However, different code formats, styles and languages of software codes and other means of configuring code to perform the tasks in accordance with the invention will not depart from the spirit and scope of the invention.

[0329] The invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described examples are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

CLAIMS:

1. A method of decoding colour pictures using coding tools including one or more cross component models related modes, the method comprising:

receiving coded data associated with a current block comprising a first-colour block and a second-colour block, wherein the current block is coded using CCP (Cross-Component Prediction);

parsing coded syntax data corresponding to one or more syntaxes related to transform coding or residual coding applied to the current block;

decoding the coded syntax data to generate said one or more syntaxes by using one or more context models according to information associated with the CCP; and

decoding the current block using the CCP, wherein said decoding comprises the transform coding of the current block or the residual coding of the current block using said one or more syntaxes.

2. The method of Claim 1, wherein said one or more syntaxes are from a group comprising root_cbf, cb_cbf, cr_cbf, transform_skip_flag, lfnst_idx, mts_idx, tu_joint_cbcr_residual_flag, last_significant_position, cu_qp_delta_abs, cu_qp_delta_sign_flag, cu_chroma_qp_offset_flag, cu_chroma_qp_offset_idx, or a combination thereof.

3. The method of Claim 1, wherein said one or more context models correspond to multiple context models and selection among the multiple context models depends on CCP type of the current block.

4. The method of Claim 1, wherein said one or more context models correspond to multiple context models and selection among the multiple context models depends on a prediction mode of a neighbouring block of the current block.

5. The method of Claim 4, wherein the prediction mode of the neighbouring block corresponds to one CCP mode.

6. The method of Claim 1, wherein said one or more context models correspond to multiple context models and selection among the multiple context models depends on a number of neighbouring blocks coded in a CCP mode.

7. The method of Claim 1, wherein said one or more context models correspond to CCP single model, CCP multiple model, CCCM (Convolutional Cross-Component Mode), GLM (Gradient Linear Model), or a combination thereof.

8. An apparatus for video decoding, the apparatus comprising one or more electronics or processors arranged to:

receive coded data associated with a current block comprising a first-colour block and a second-colour block, wherein the current block is coded using CCP (Cross-Component Prediction);

parse coded syntax data corresponding to one or more syntaxes related to transform coding or residual coding applied to the current block;

decode the coded syntax data to generate said one or more syntaxes by using one or more context models according to information associated with the CCP; and

decode the current block using the CCP, wherein decoding process of the current block comprises the transform coding of the current block or the residual coding of the current block using said one or more syntaxes.

9. A method of encoding colour pictures using coding tools including one or more cross component models related modes, the method comprising:

receiving input data associated with a current block comprising a first-colour block and a second-colour block;

encoding the current block using CCP (Cross-Component Prediction), wherein said encoding comprises transform coding to generate transformed data of the current block or residual coding to generate residual data of the current block; and

encoding one or more syntaxes related to the transform coding or the residual coding to generate coded syntax data by using one or more context models according to information associated with the CCP; and

signalling the coded syntax data.

10. An apparatus for video encoding, the apparatus comprising one or more electronics or processors arranged to:

receive input data associated with a current block comprising a first-colour block and a second-colour block;

encode the current block by using CCP (Cross-Component Prediction), wherein encoding process of the current block comprises transform coding to generate transformed data of the current block or residual coding to generate residual data of the current block; and

encode one or more syntaxes related to the transform coding or the residual coding to generate coded syntax data by using one or more context models according to information associated with the CCP; and

signal the coded syntax data.

11. A method of coding colour pictures using coding tools including one or more cross component

models related modes, the method comprising:

receiving input data associated with a current block comprising a luma block and two or more chroma blocks, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction); and

encoding or decoding the current block by using the CCP, wherein said encoding or decoding uses information comprising a chroma syntax corresponding to one or more coded block flags comprising a root chroma coded block flag.

12. The method of Claim 11, wherein an inherited CCP candidate from a neighbouring block is included in a merge candidate list for the current block and a merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not.

13. The method of Claim 11, wherein said one or more coded block flags further comprise two or more component coded block flags corresponding to said two or more chroma blocks.

14. The method of Claim 13, wherein if the root chroma coded block flag is false, each of said two or more component coded block flags corresponding to said two or more chroma blocks is equal to 0 or if the root chroma coded block flag is true, at least one of said two or more component coded block flags corresponding to said two or more chroma blocks is equal to 1.

15. An apparatus for video coding, the apparatus comprising one or more electronics or processors arranged to:

receive input data associated with a current block comprising a luma block and two or more chroma blocks, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction); and

encode or decode the current block by using the CCP, wherein encoding or decoding process of the current block uses information comprising a syntax corresponding to one or more coded block flags comprising a root chroma coded block flag.

16. The apparatus of Claim 15, wherein said one or more coded block flags further comprise two or more component coded block flags corresponding to said two or more chroma blocks.

17. The apparatus of Claim 16, wherein if the root chroma coded block flag is false, each said two or more component coded block flags corresponding to the said two or more chroma blocks is equal

to 0, or if the root chroma coded block flag is true, at least one of said two or more component coded block flags corresponding to said two or more chroma blocks is equal to 1.

18. A method of coding colour pictures using coding tools including one or more cross component models related modes, the method comprising:

receiving input data associated with a current block comprising a luma block and two or more chroma blocks, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction);

generating a merge candidate list for the current block, wherein the merge candidate list includes an inherited CCP candidate from a neighbouring block;

signalling or parsing a merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not;

signalling or parsing a skip flag before the merge flag, wherein when the skip flag is true, the merge flag is inferred; and

encoding or decoding the current block by using information comprising the merge candidate list.

19. The method of Claim 18, wherein when the skip flag is true, an index associated with the inherited CCP candidate is signalled or parsed.

20. The method of Claim 18, wherein when the merge flag is inferred to be 1, the current block inherits said one or more cross-component model parameters from the inherited CCP candidate.

21. The method of Claim 18, wherein when the merge flag is inferred to be 0, a CCP mode index is signalled or parsed.

22. The method of Claim 18, wherein a high-level syntax is signalled or parsed in SPS (Sequence Parameter Set), PPS (Picture Parameter Set), PH (Picture header) or SH (Slice header) to indicate whether the skip flag is allowed for a current sequence, picture, or slice.

23. An apparatus for video coding, the apparatus comprising one or more electronics or processors arranged to:

receive input data associated with a current block comprising a luma block and two or more chroma blocks, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction);

generate a merge candidate list for the current block, wherein the merge candidate list includes an inherited CCP candidate from a neighbouring block;

signal or parse a merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not;

signal or parse a skip flag before the merge flag, wherein when the skip flag is true, the merge flag is inferred; and

encode or decode the current block by using information comprising the merge candidate list.

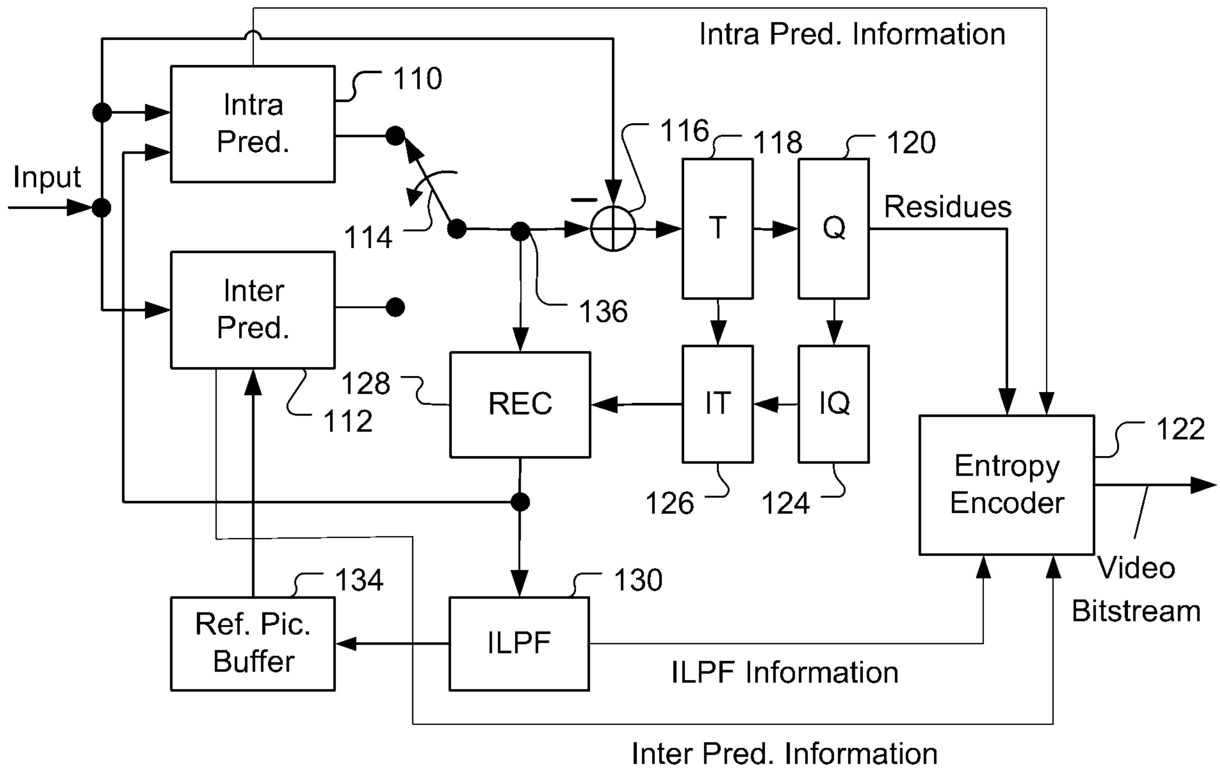
Intra Pred. Information
Intra Pred.
110
Input
116
118
120
T
Q
Residues
114
136
Inter Pred.
128
REC
IT
IQ
112
Entropy Encoder
122
126
124
Video Bitstream
134
130
Ref. Pic. Buffer
ILPF
ILPF Information
Inter Pred. Information

Fig. 1A

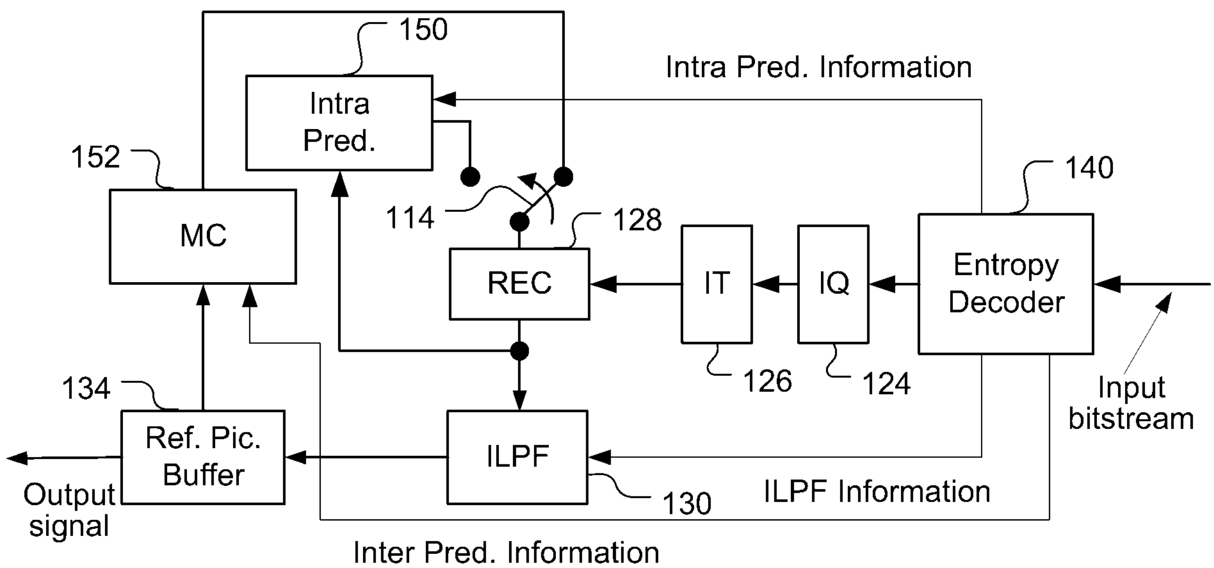
150
Intra Pred. Information
Intra Pred.
152
140
MC
114
128
REC
IT
IQ
Entropy Decoder
134
126
124
Input bitstream
Ref. Pic. Buffer
ILPF
Output signal
130
ILPF Information
Inter Pred. Information

Fig. 1B

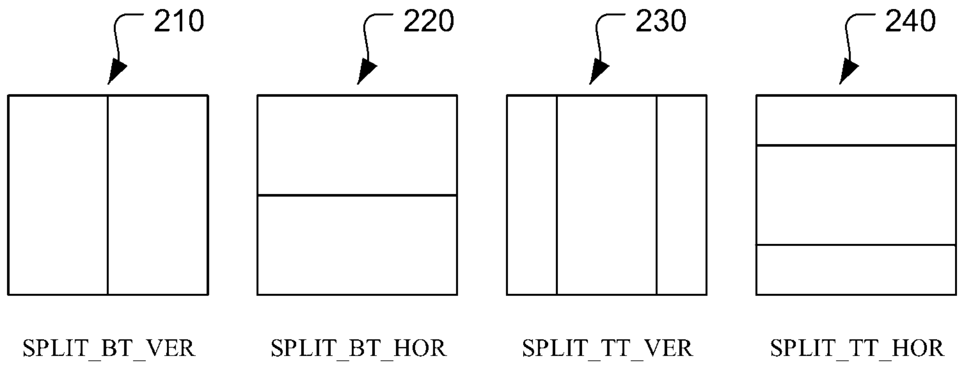
210
220
230
240
SPLIT_BT_VER
SPLIT_BT_HOR
SPLIT_TT_VER
SPLIT_TT_HOR

Fig. 2

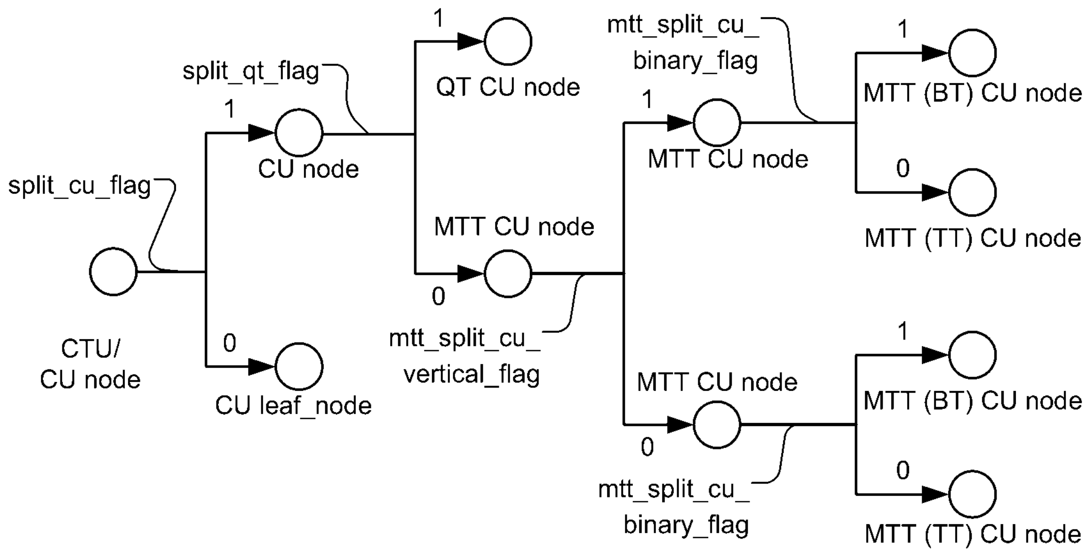
split_cu_flag
CTU/
CU node
1
0
CU node
CU leaf_node
split_qt_flag
1
QT CU node
MTT CU node
0
mtt_split_cu_
vertical_flag
mtt_split_cu_
binary_flag
1
MTT CU node
MTT CU node
0
mtt_split_cu_
binary_flag
1
MTT (BT) CU node
0
MTT (TT) CU node
1
MTT (BT) CU node
0
MTT (TT) CU node

Fig. 3

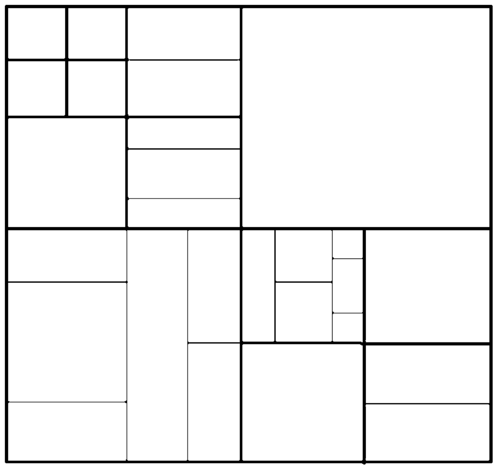

Fig. 4

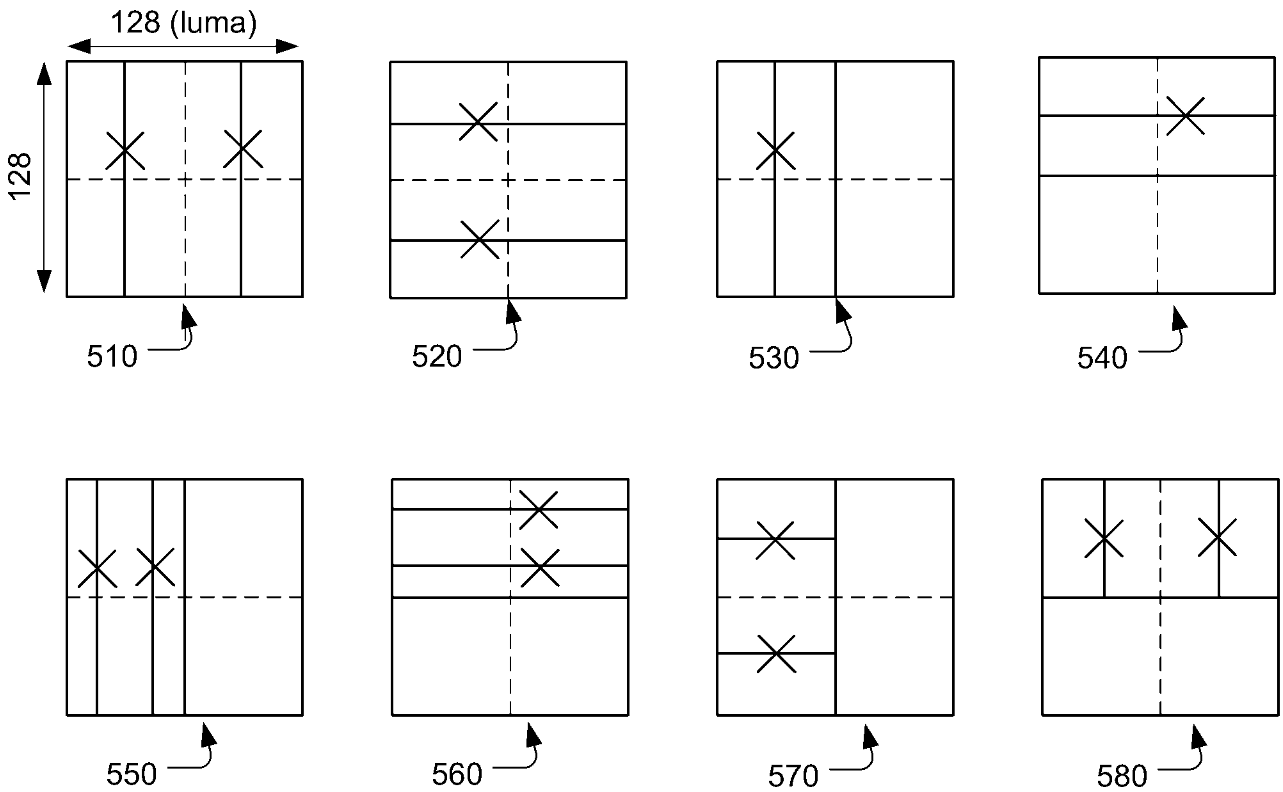

Fig. 5

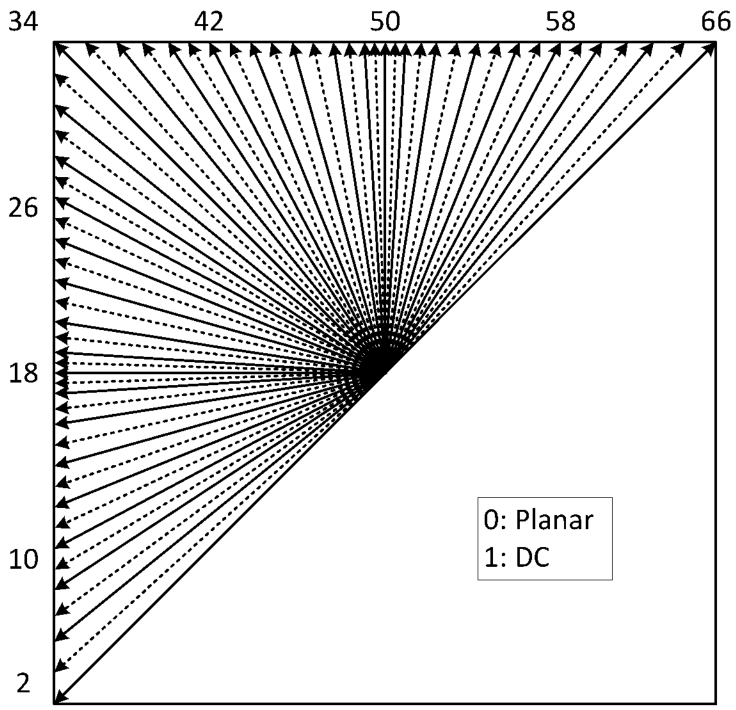
34
42
50
58
66
26
18
10
2
0: Planar
1: DC

Fig. 6

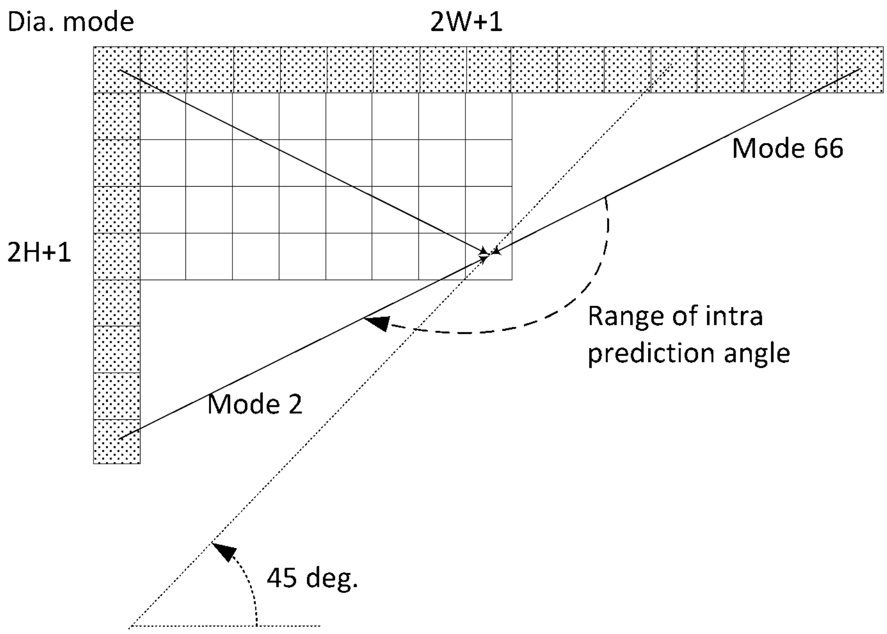
Dia. mode
2W+1
Mode 66
2H+1
Range of intra prediction angle
Mode 2
45 deg.

Fig. 7A

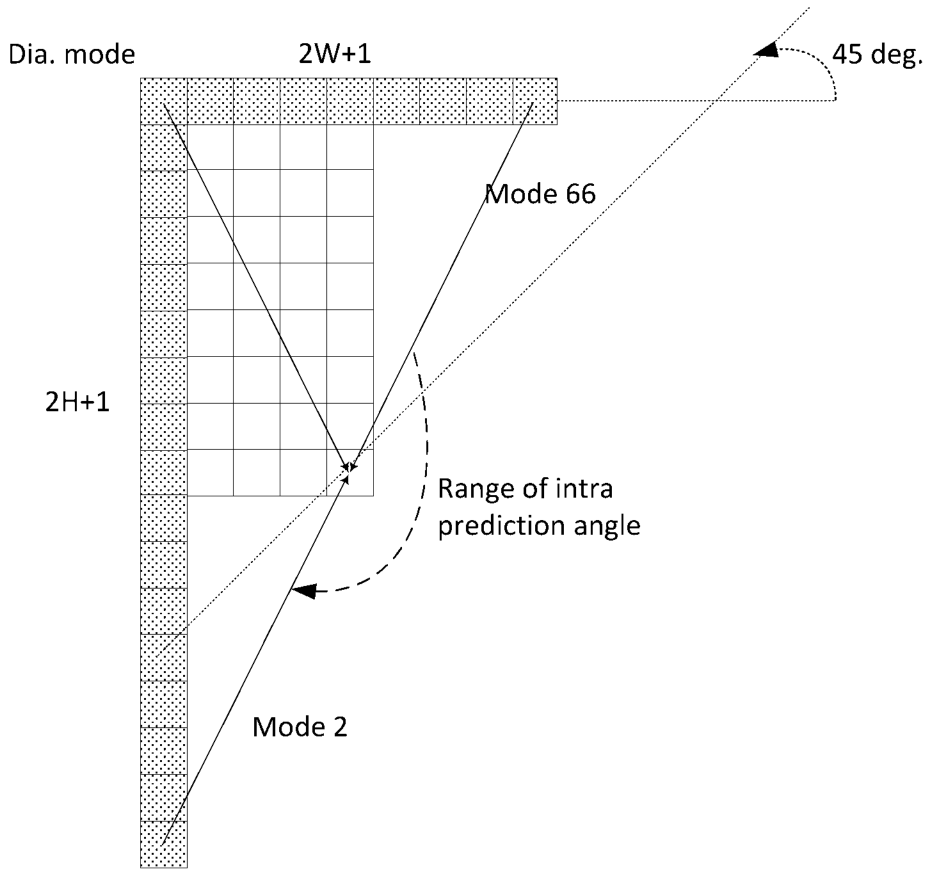
Dia. mode
2W+1
45 deg.
Mode 66
2H+1
Range of intra
prediction angle
Mode 2

Fig. 7B

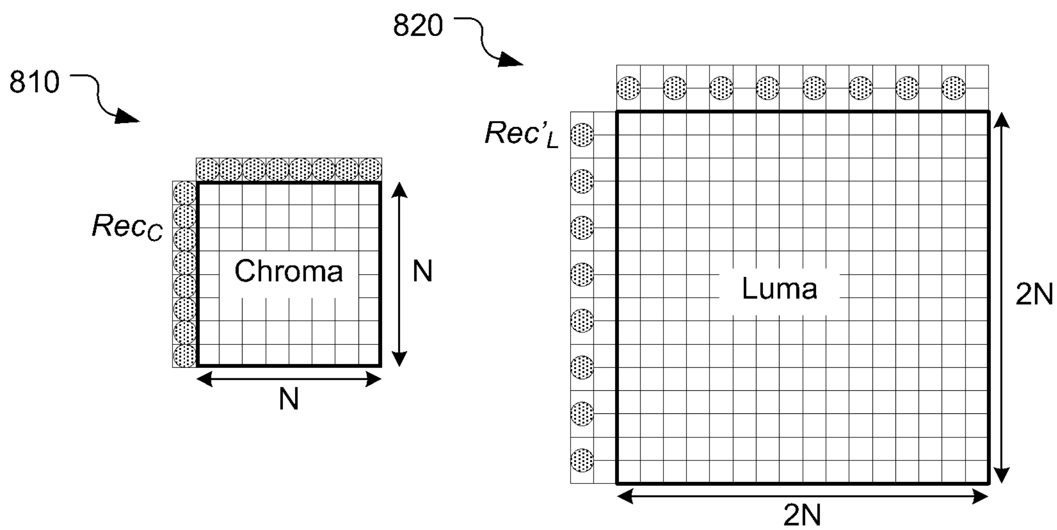
820
810
Rec'L
RecC
Chroma
N
N
Luma
2N
2N

Fig. 8

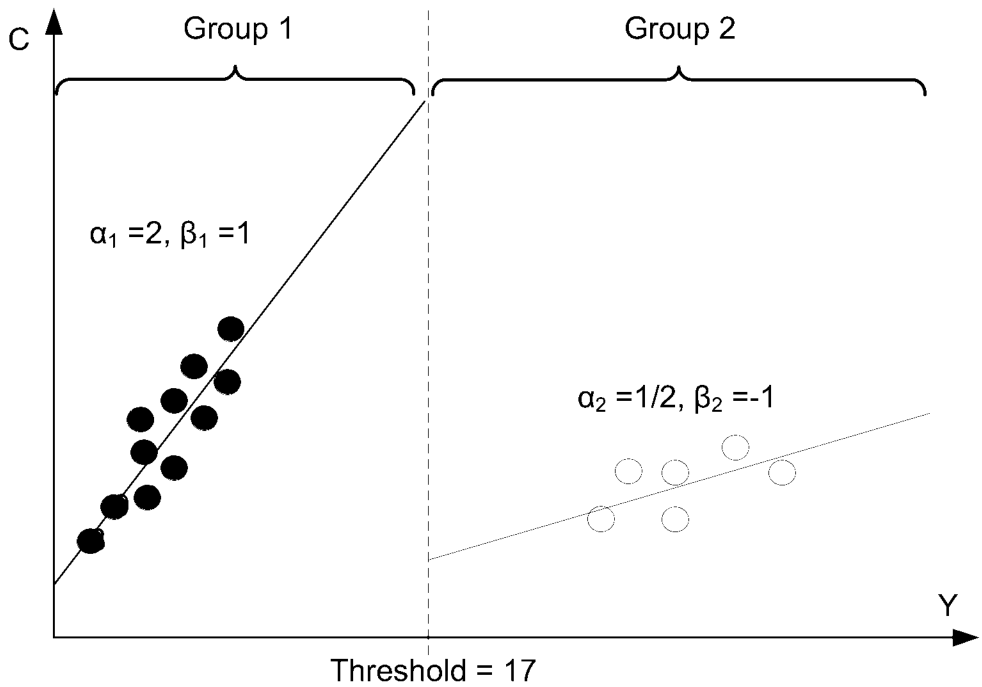

Fig. 9

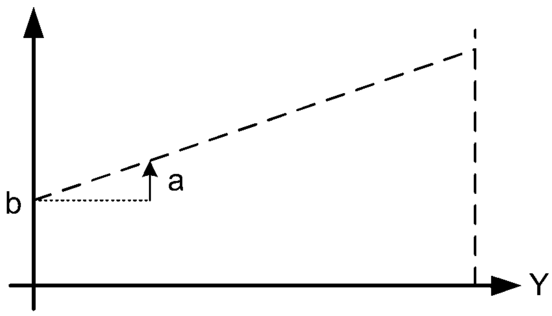

Fig. 10A

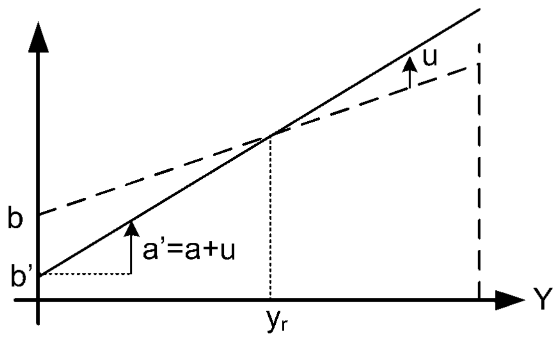

Fig. 10B

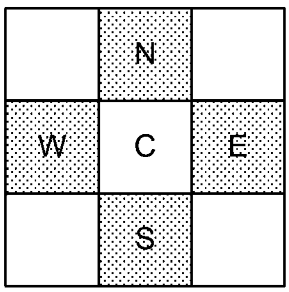

Fig. 11

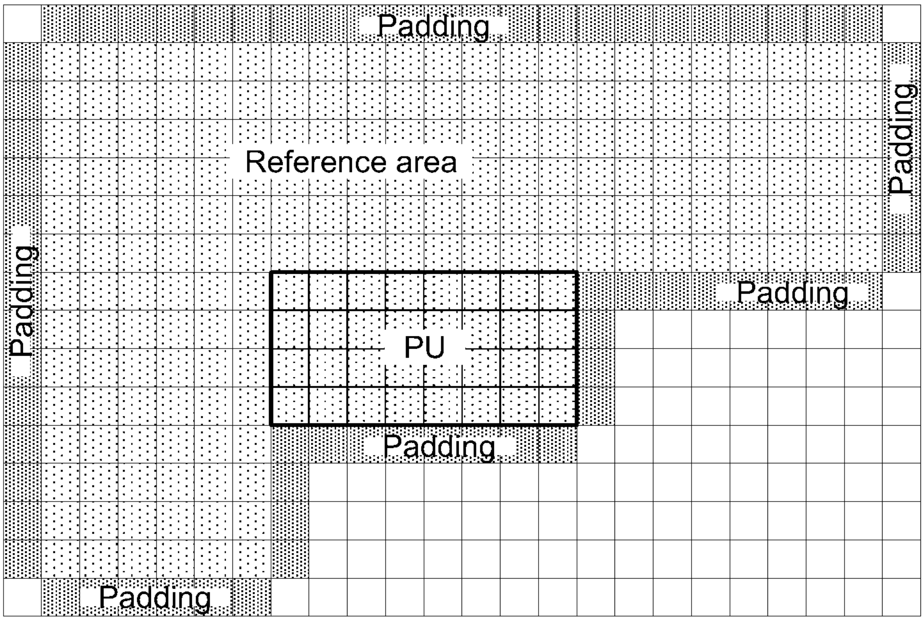
Padding
Padding
Reference area
Padding
Padding
PU
Padding
Padding

Fig. 12

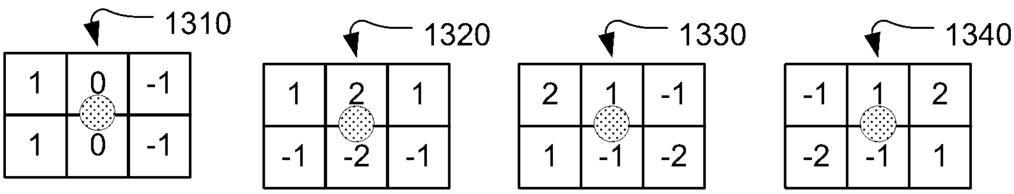
1310
1
0
-1
1
0
-1
1320
1
2
1
-1
-2
-1
1330
2
1
-1
1
-1
-2
1340
-1
1
2
-2
-1
1

Fig. 13

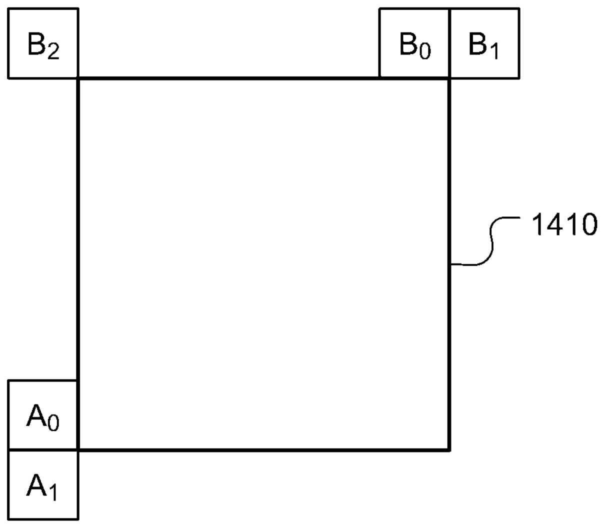

Fig. 14

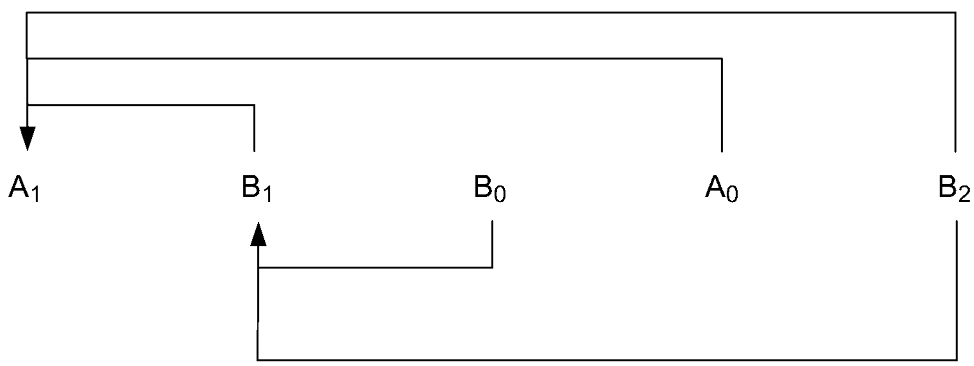

Fig. 15

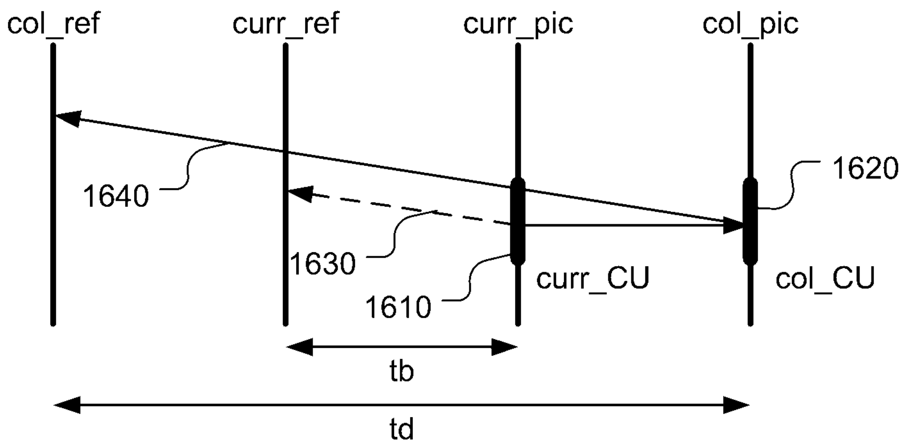

Fig. 16

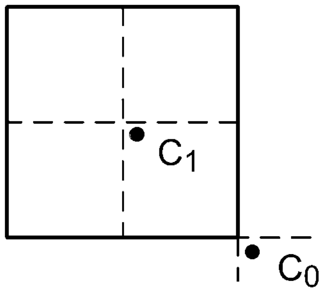

Fig. 17

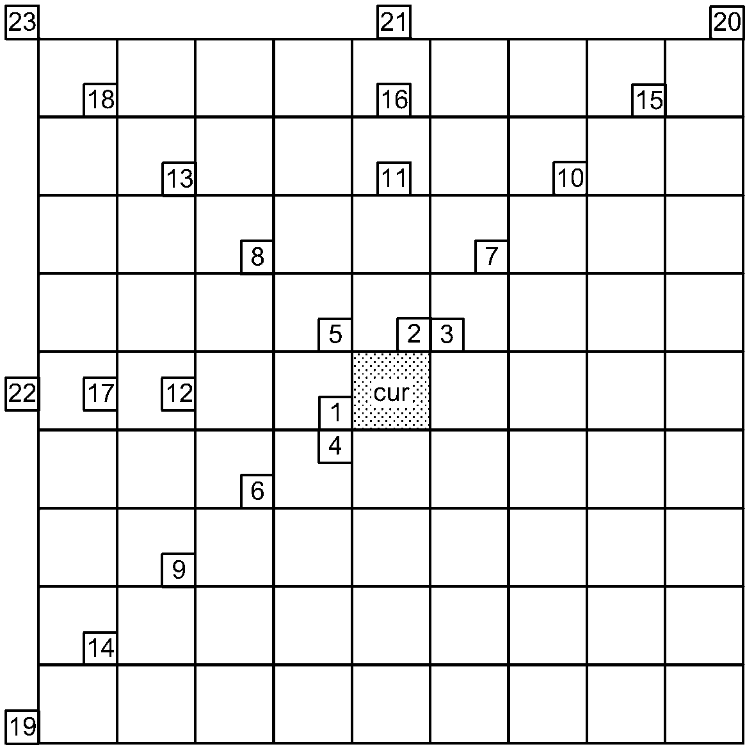

Fig. 18

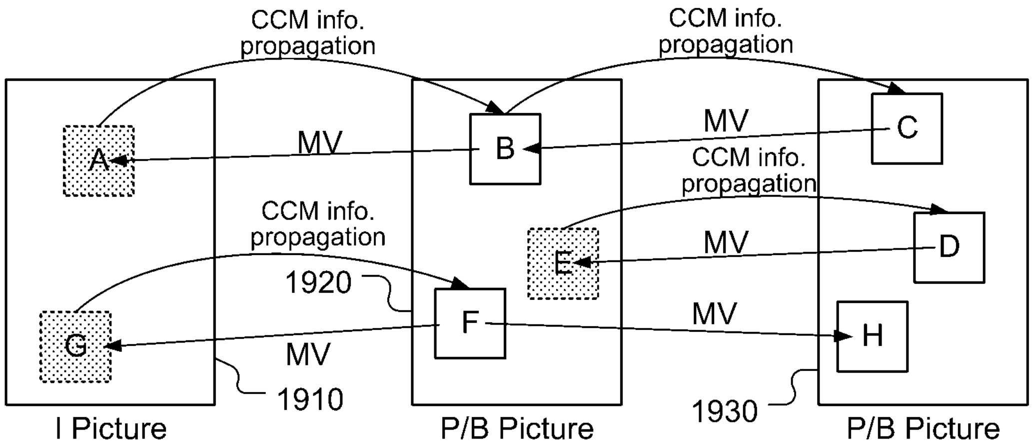

Fig. 19

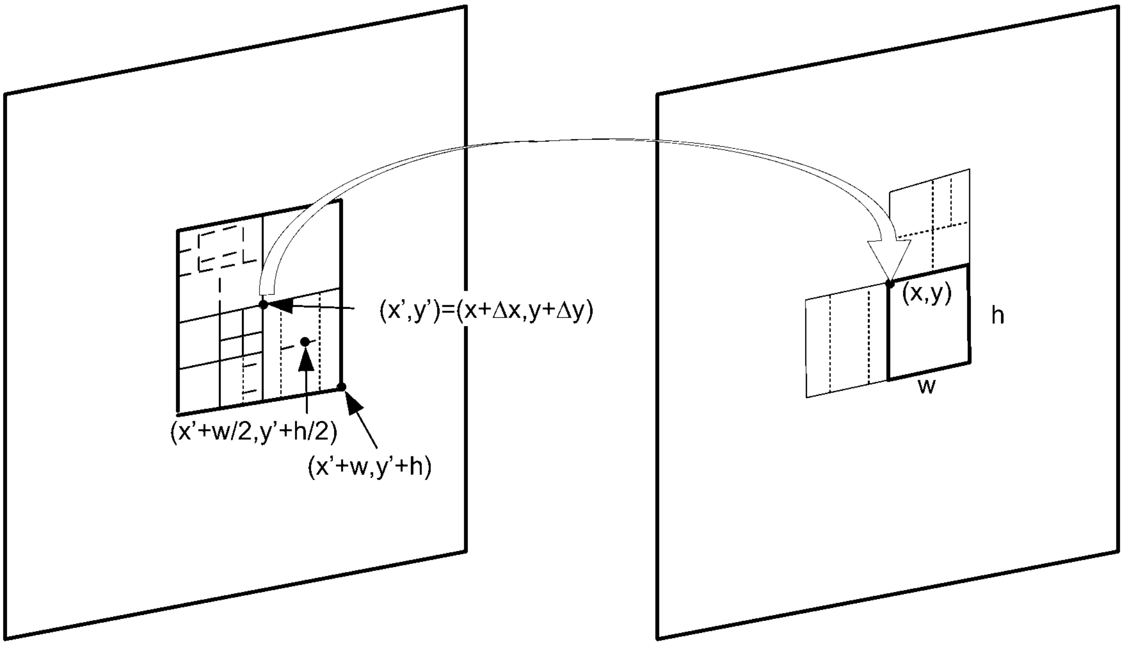

Fig. 20

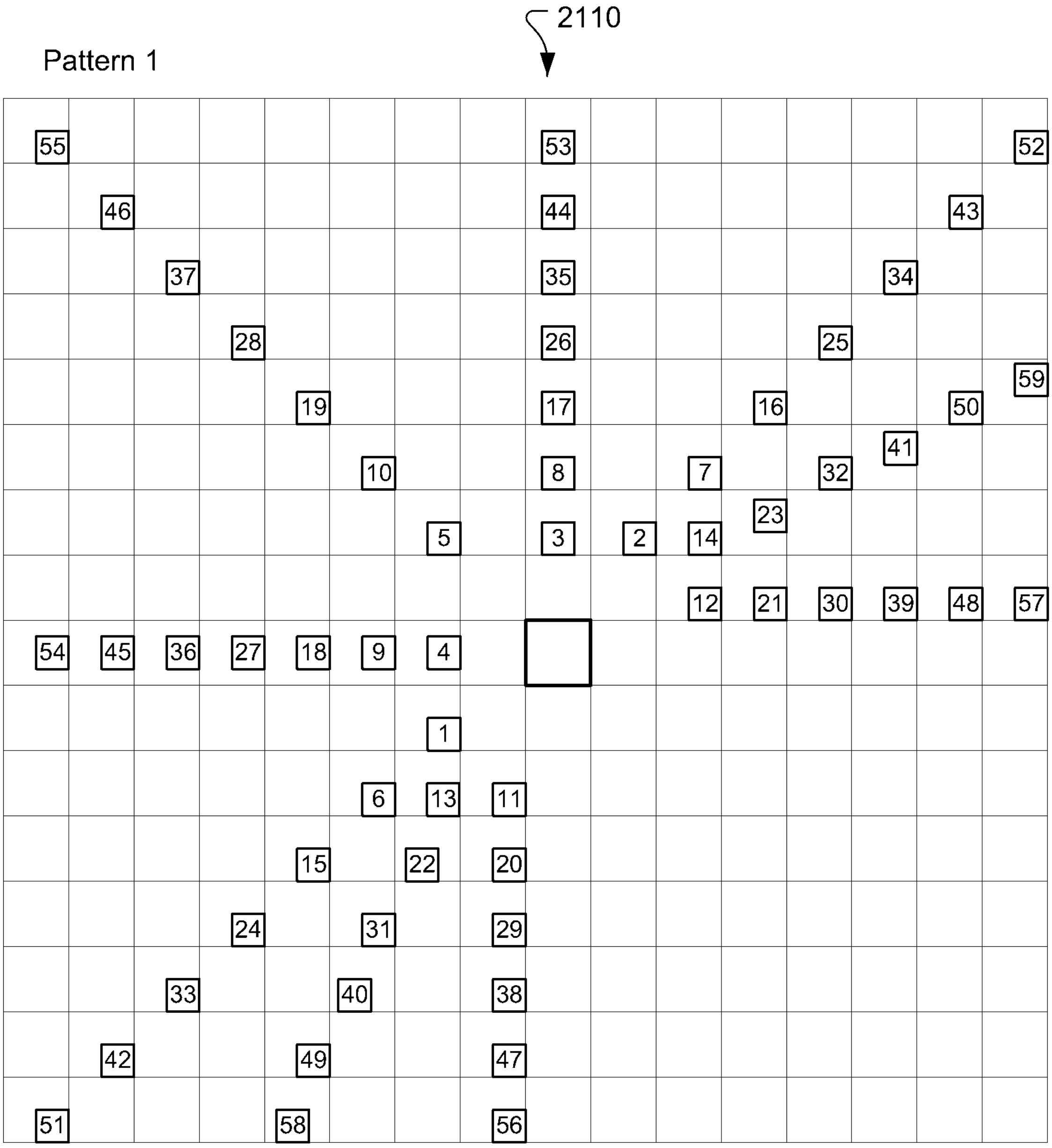
2110
Pattern 1
55
53
52
46
44
43
37
35
34
28
26
25
59
19
17
16
50
41
10
8
7
32
23
5
3
2
14
12
21
30
39
48
57
54
45
36
27
18
9
4
1
6
13
11
15
22
20
24
31
29
33
40
38
42
49
47
51
58
56

Fig. 21A

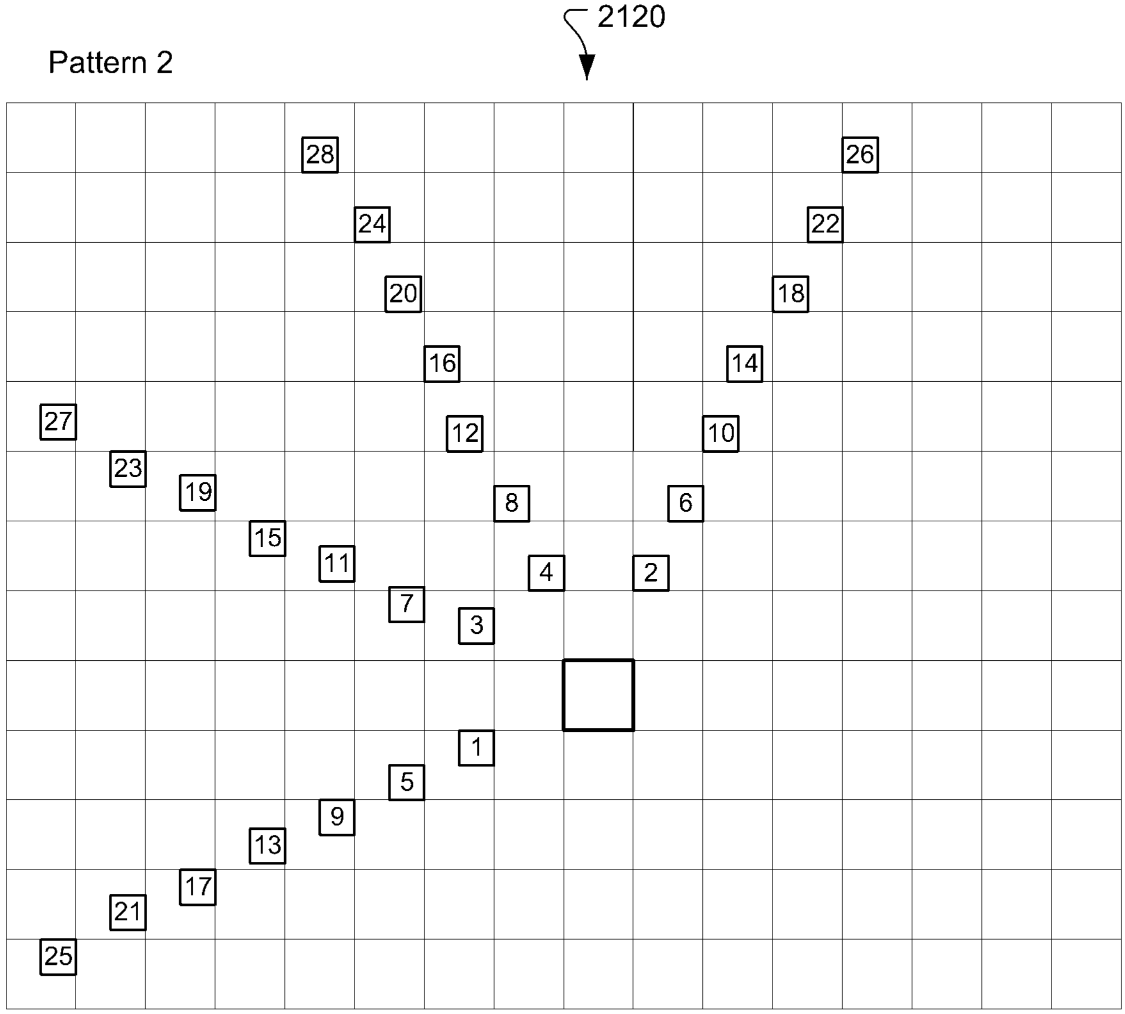
2120
Pattern 2
28
26
24
22
20
18
16
14
27
12
10
23
19
8
6
15
11
4
2
7
3
1
5
9
13
17
21
25

Fig. 21B

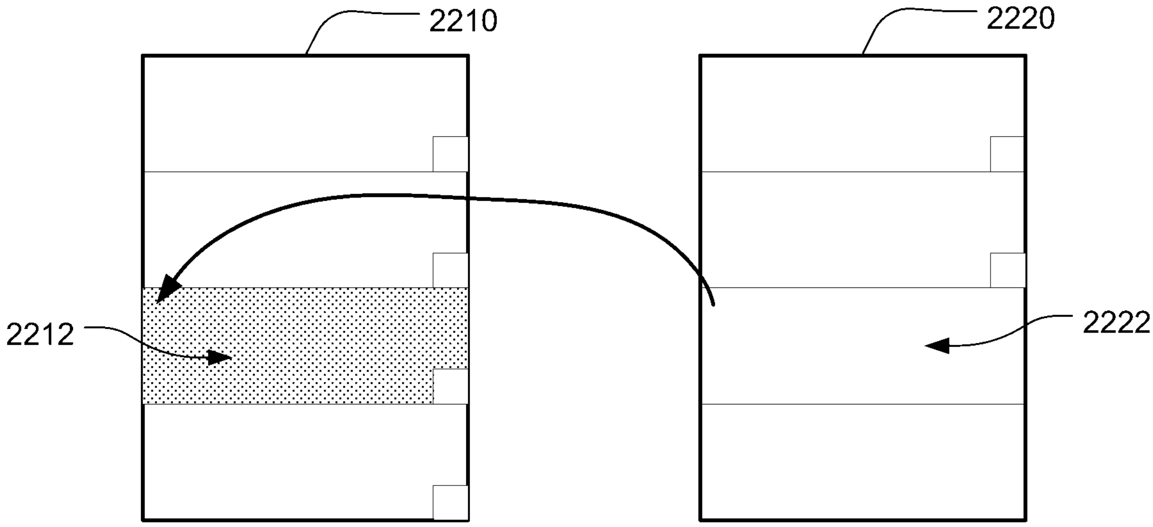

Fig. 22A

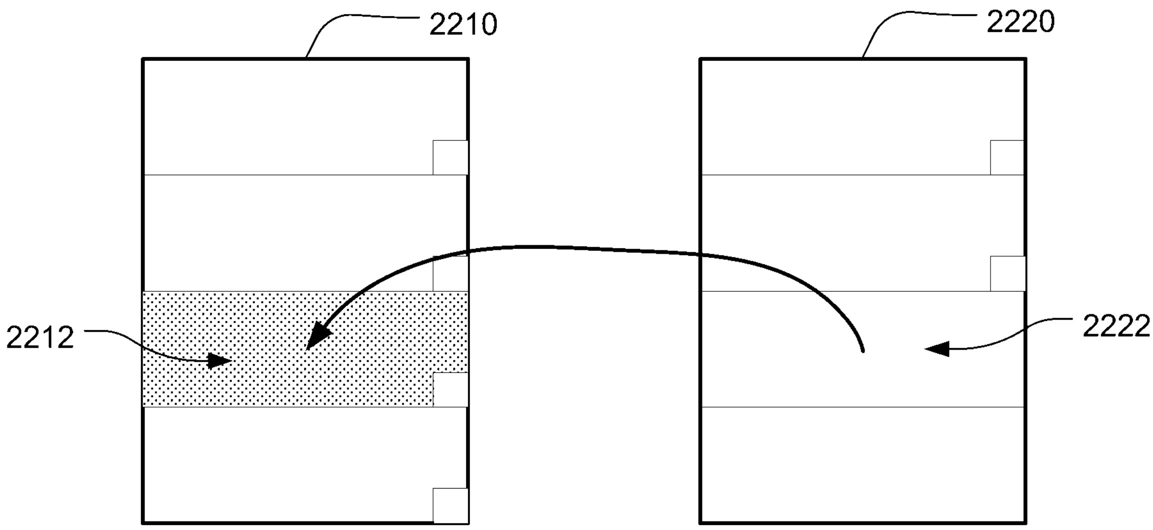

Fig. 22B

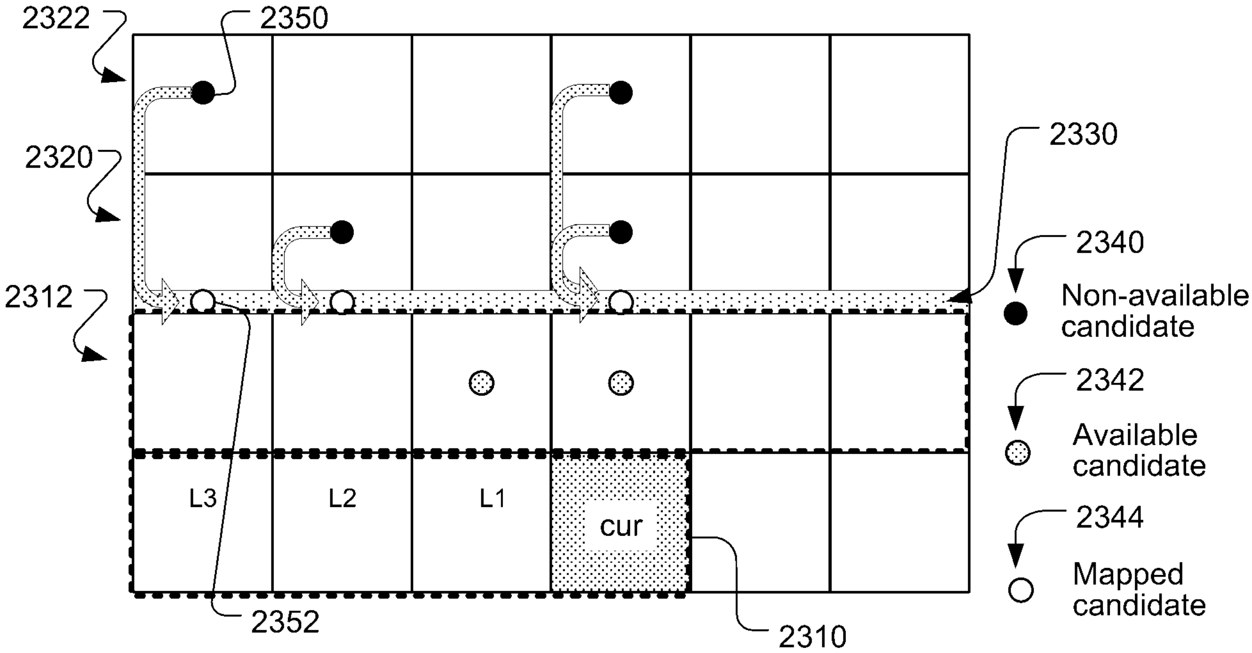
2322
2350
2320
2330
2340
Non-available candidate
2312
2342
Available candidate
L3
L2
L1
cur
2344
Mapped candidate
2352
2310

Fig. 23

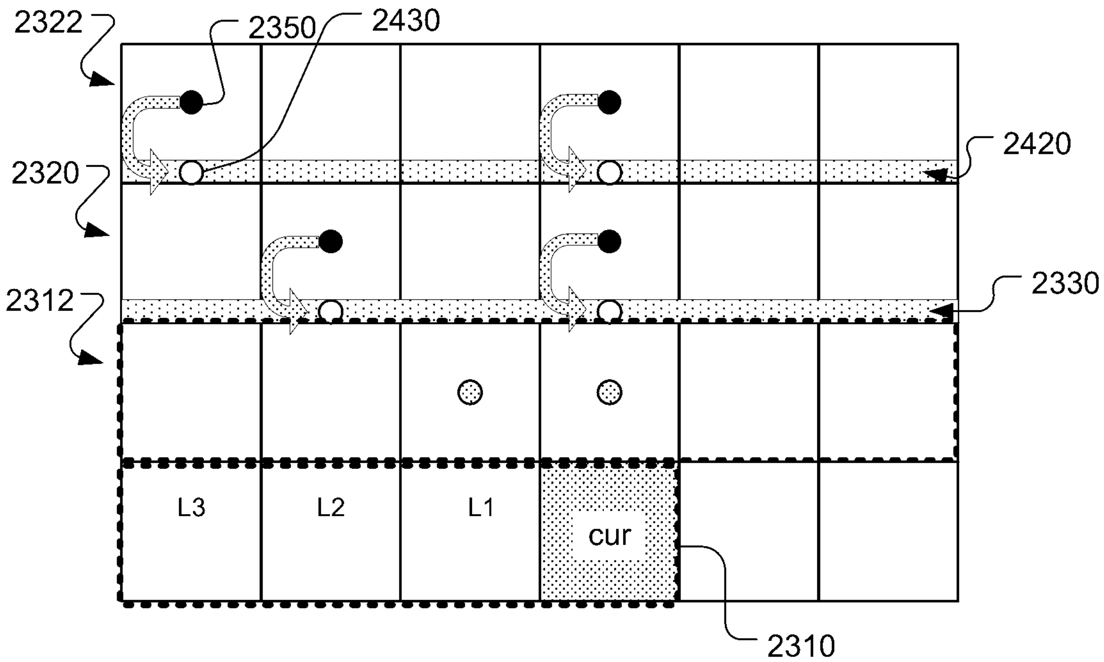
2322
2350
2430
2420
2320
2312
2330
L3
L2
L1
cur
2310

Fig. 24

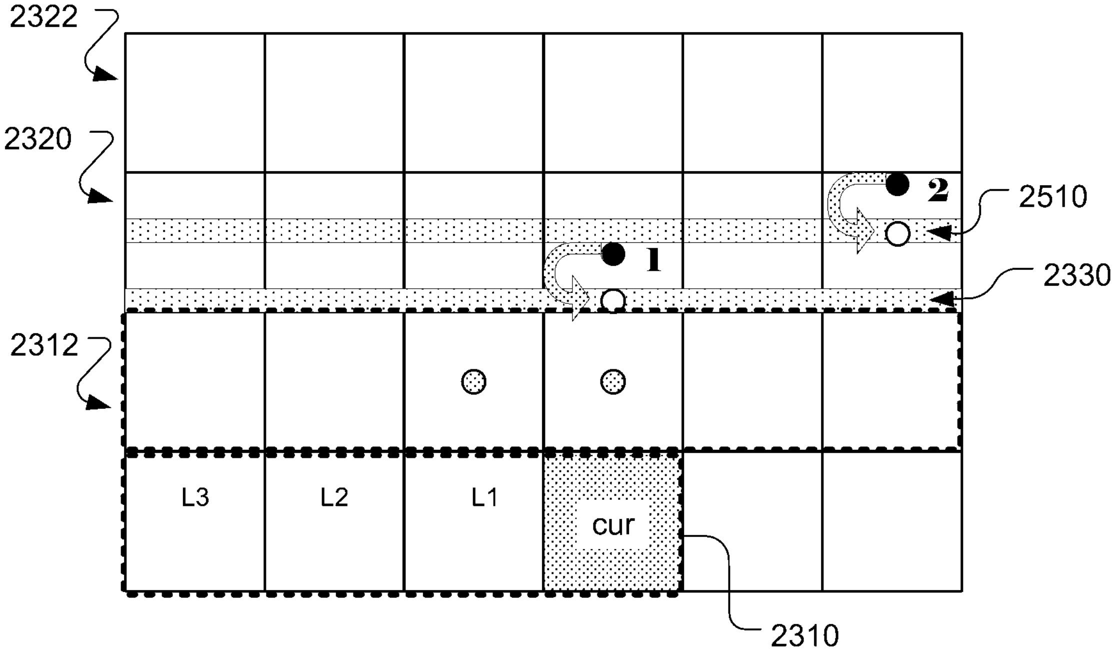
2322
2320
2510
2330
2312
1
2
L3
L2
L1
cur
2310

Fig. 25

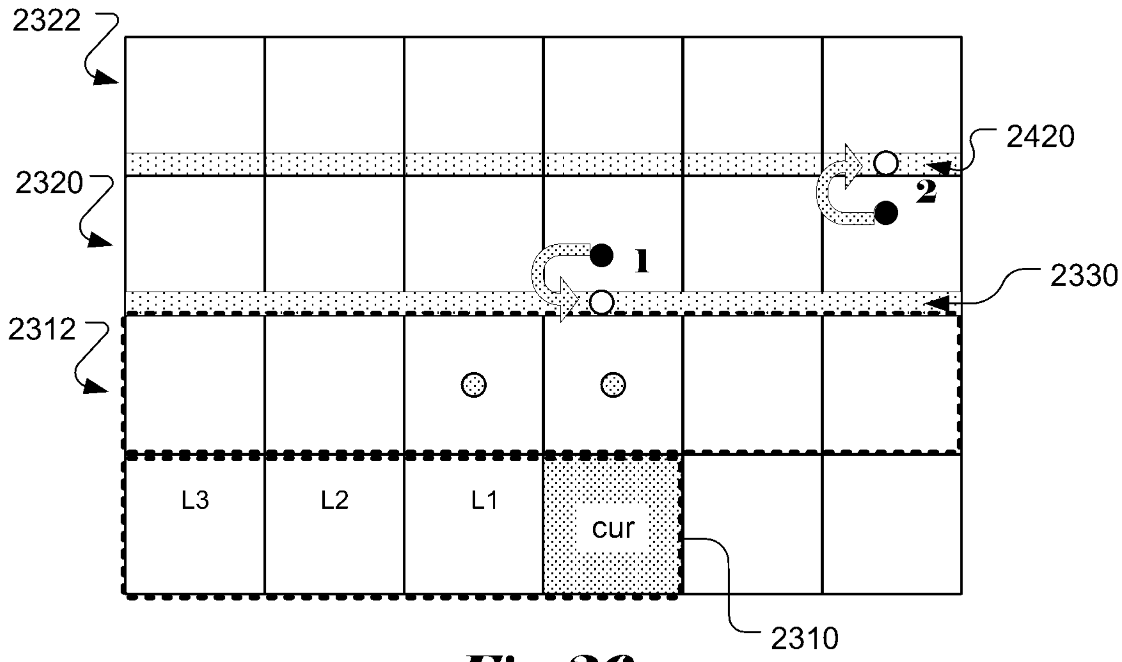
2322
2420
2320
2330
2312
1
2
L3
L2
L1
cur
2310

Fig. 26

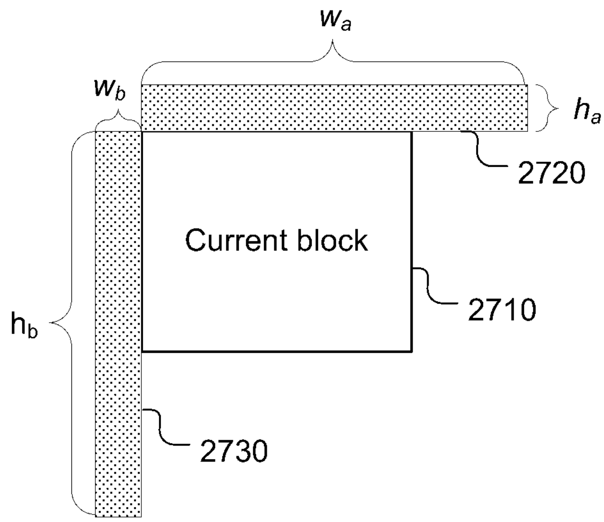
w_a
w_b
h_a
2720
Current block
2710
h_b
2730

Fig. 27

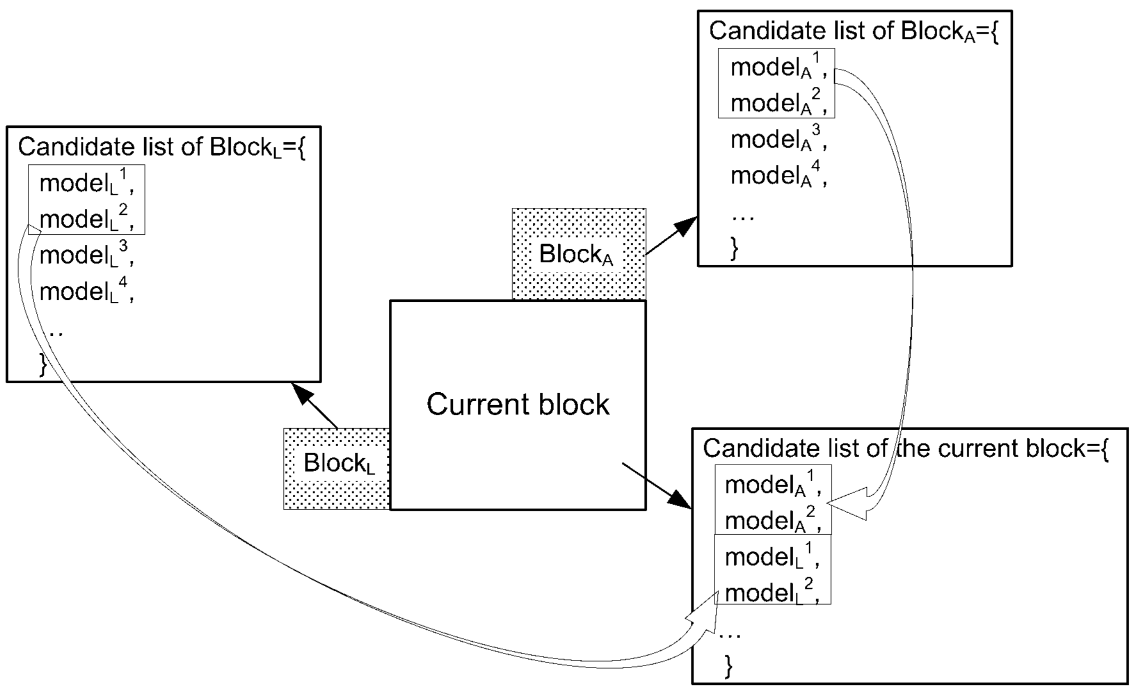
Candidate list of $Block_A$={
$model_A^1$,
$model_A^2$,
$model_A^3$,
$model_A^4$,
…
}
Candidate list of $Block_L$={
$model_L^1$,
$model_L^2$,
$model_L^3$,
$model_L^4$,
..
}
$Block_A$
Current block
$Block_L$
Candidate list of the current block={
$model_A^1$,
$model_A^2$,
$model_L^1$,
$model_L^2$,
…
}

Fig. 28

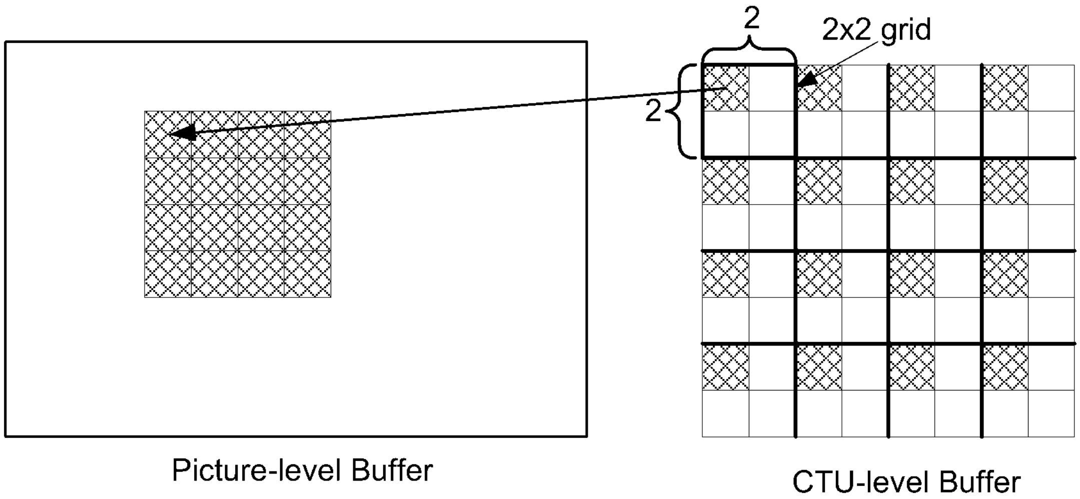

Fig. 29

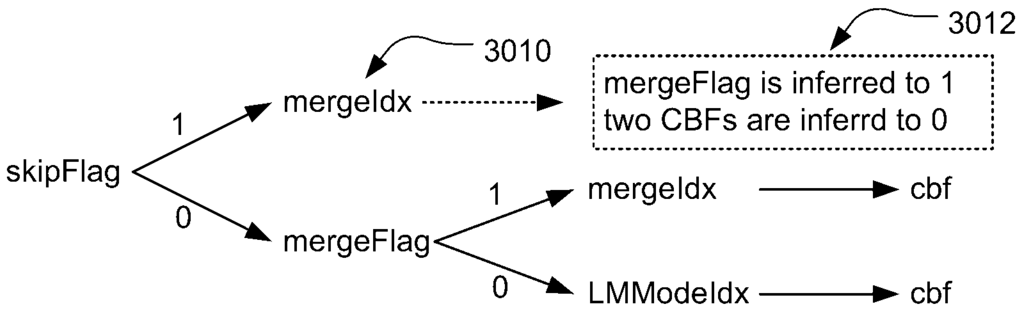

Fig. 30

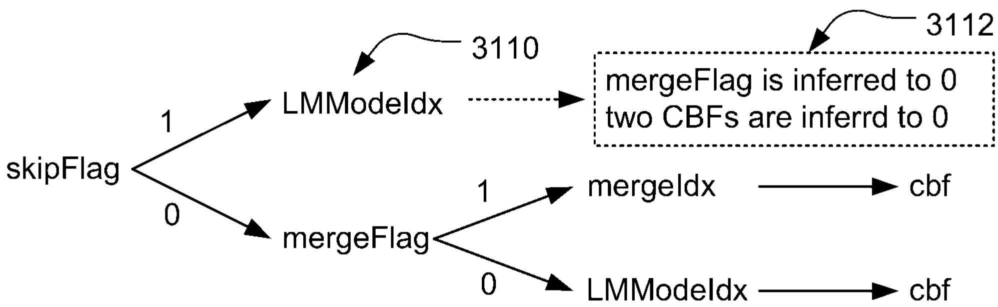

Fig. 31

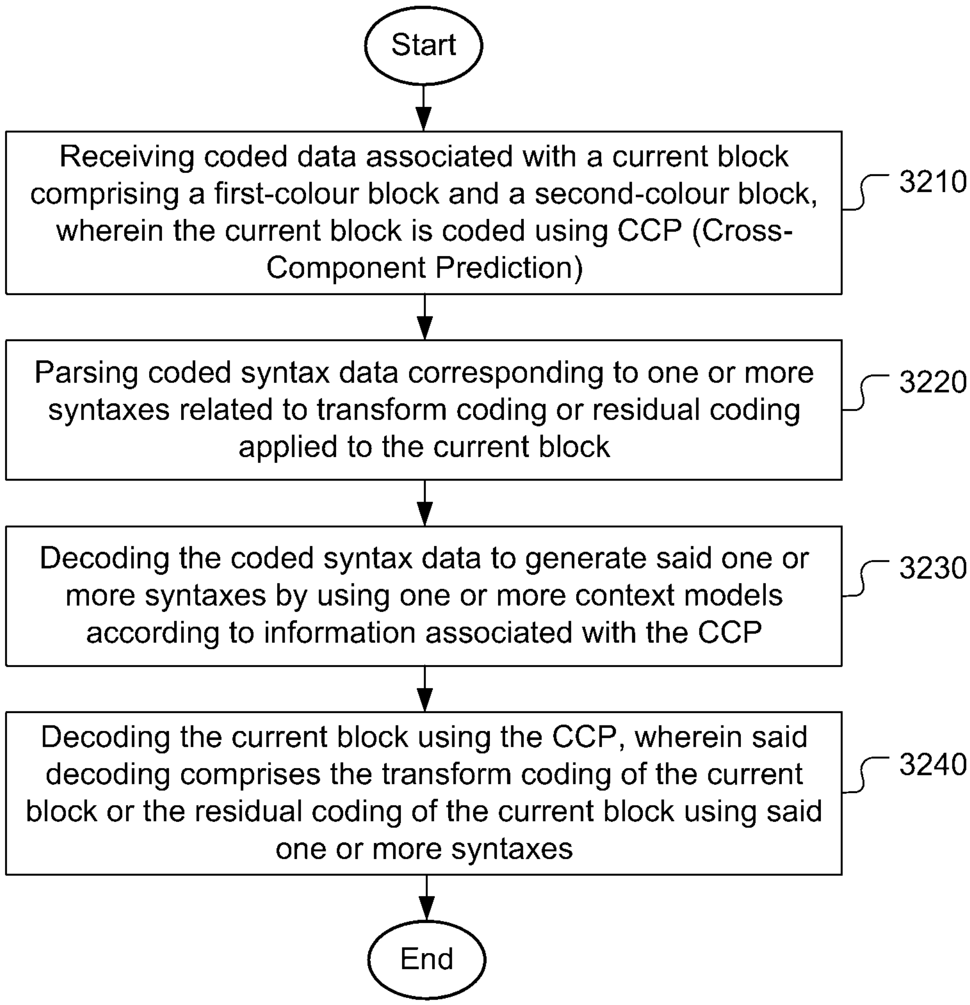
Start
Receiving coded data associated with a current block comprising a first-colour block and a second-colour block, wherein the current block is coded using CCP (Cross-Component Prediction)
3210
Parsing coded syntax data corresponding to one or more syntaxes related to transform coding or residual coding applied to the current block
3220
Decoding the coded syntax data to generate said one or more syntaxes by using one or more context models according to information associated with the CCP
3230
Decoding the current block using the CCP, wherein said decoding comprises the transform coding of the current block or the residual coding of the current block using said one or more syntaxes
3240
End

Fig. 32

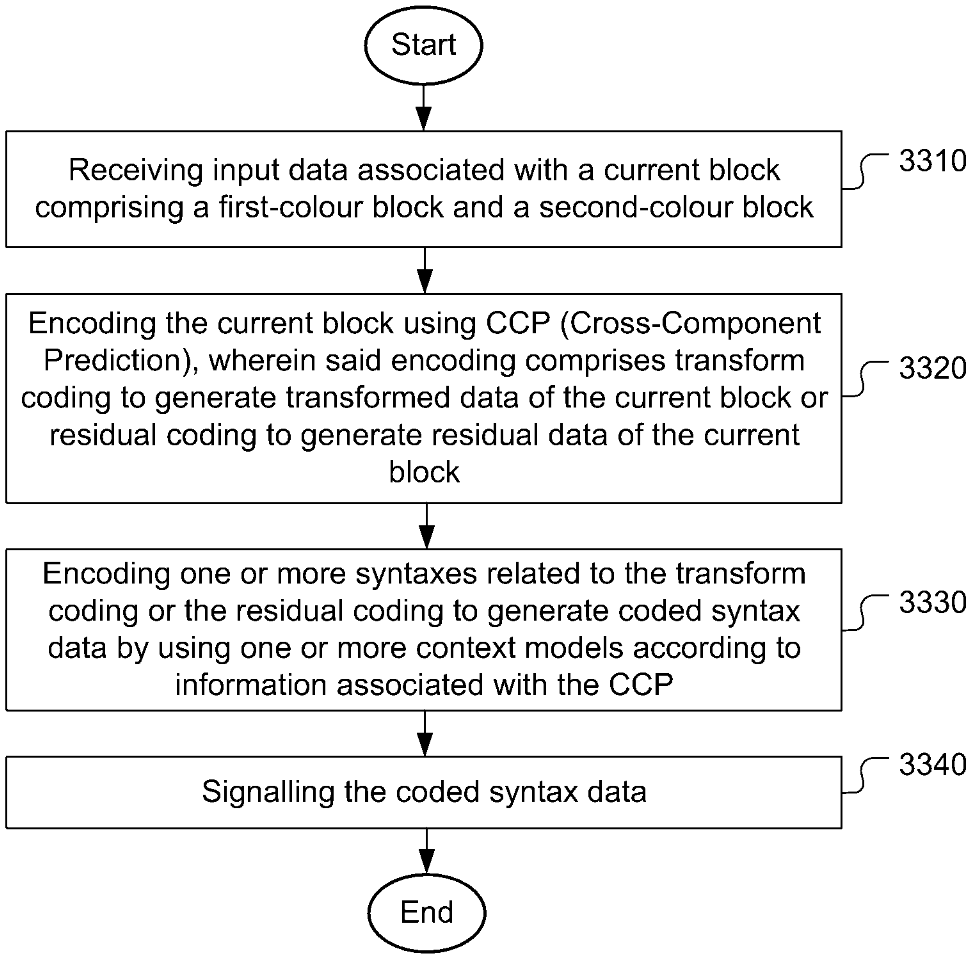
Start
Receiving input data associated with a current block comprising a first-colour block and a second-colour block
3310
Encoding the current block using CCP (Cross-Component Prediction), wherein said encoding comprises transform coding to generate transformed data of the current block or residual coding to generate residual data of the current block
3320
Encoding one or more syntaxes related to the transform coding or the residual coding to generate coded syntax data by using one or more context models according to information associated with the CCP
3330
Signalling the coded syntax data
3340
End

Fig. 33

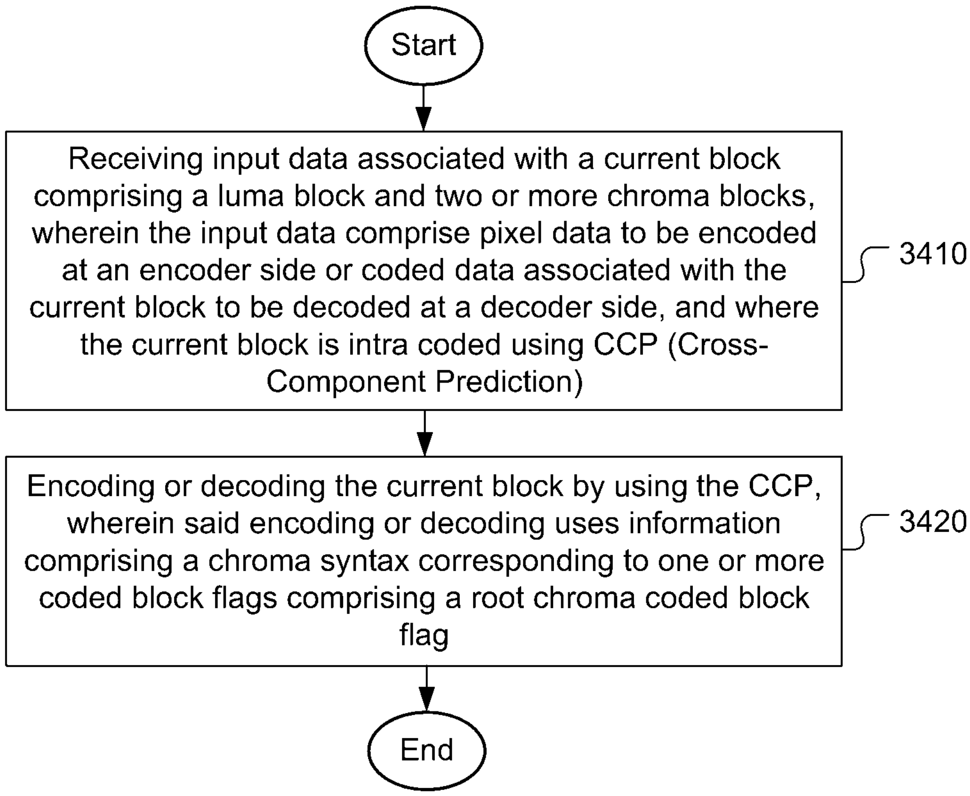
Start
Receiving input data associated with a current block comprising a luma block and two or more chroma blocks, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction)
3410
Encoding or decoding the current block by using the CCP, wherein said encoding or decoding uses information comprising a chroma syntax corresponding to one or more coded block flags comprising a root chroma coded block flag
3420
End

Fig. 34

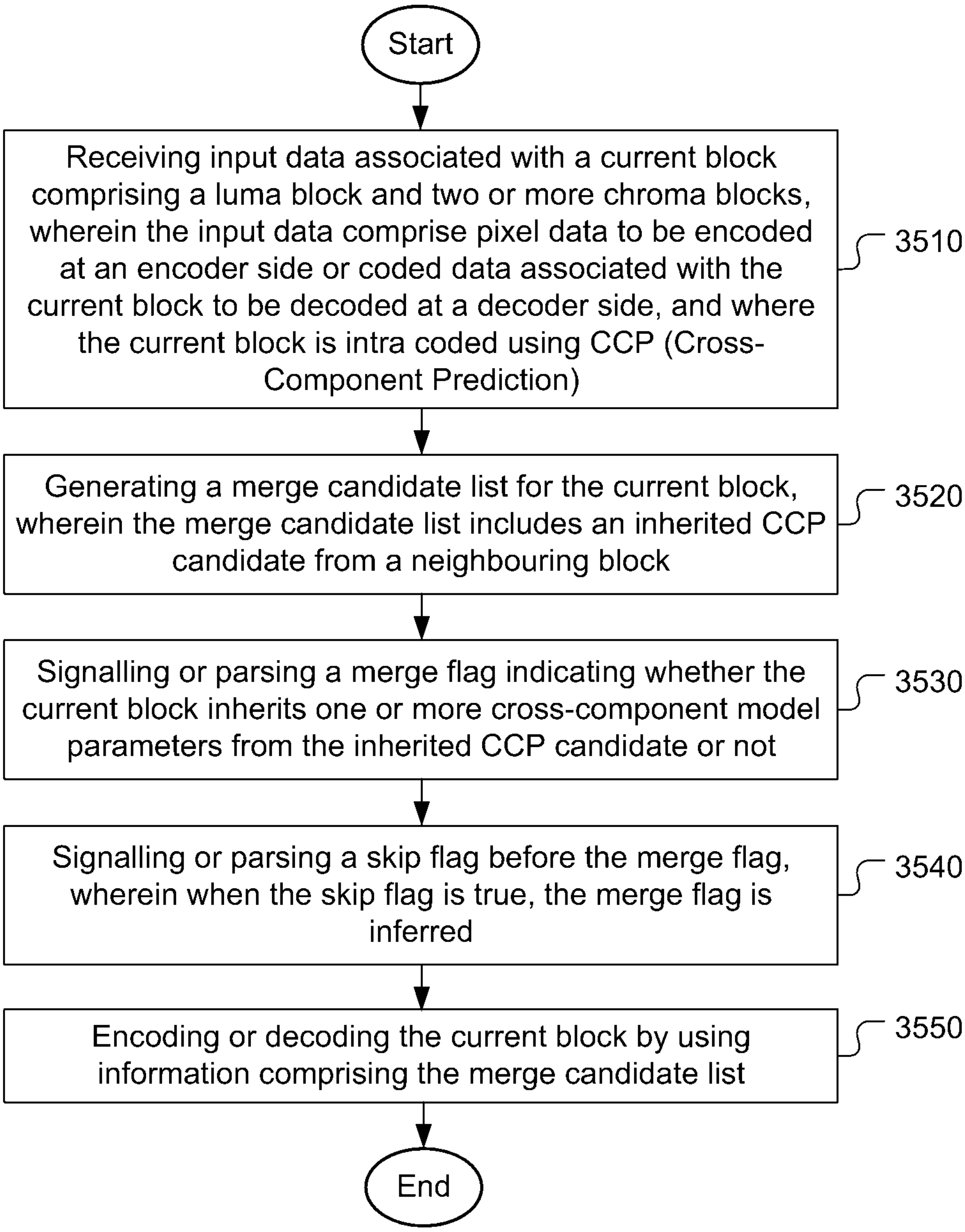
Start
Receiving input data associated with a current block comprising a luma block and two or more chroma blocks, wherein the input data comprise pixel data to be encoded at an encoder side or coded data associated with the current block to be decoded at a decoder side, and where the current block is intra coded using CCP (Cross-Component Prediction)
3510
Generating a merge candidate list for the current block, wherein the merge candidate list includes an inherited CCP candidate from a neighbouring block
3520
Signalling or parsing a merge flag indicating whether the current block inherits one or more cross-component model parameters from the inherited CCP candidate or not
3530
Signalling or parsing a skip flag before the merge flag, wherein when the skip flag is true, the merge flag is inferred
3540
Encoding or decoding the current block by using information comprising the merge candidate list
3550
End

Fig. 35

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2024/071591

A. CLASSIFICATION OF SUBJECT MATTER

H04N19/159(2014.01)i; H04N19/186(2014.01)i; H04N19/176(2014.01)i; H04N19/70(2014.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC:H04N19

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)

CNKI,ENTXT,ENTXTC,WPABS,WPABSC,CNTXT, JVET: CCP,CCLM,CCCM,GLM,intra pred,luma,chroma syntax,root chroma, flag,inherit

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CN 113170122 A (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. et al.) 23 July 2021 (2021-07-23) claim 1,paragraphs 36-40、74-82、363、518-531、606、1153-1178、1240, Drawings	1-11,13-17
Y	CN 113228644 A (MEDIATEK INC.) 06 August 2021 (2021-08-06) claim 1, paragraph 17、42-78, drawings	1-11,13-17
Y	US 2021352304 A1 (HUAWEI TECH CO LTD) 11 November 2021 (2021-11-11) paragraphs 427-462,drawings	13-14,16-17
A	CN 113892267 A (BYTEDANCE LTD.) 04 January 2022 (2022-01-04) the whole document	1-23
A	CN 115002459 A (SK TELECOM CO., LTD.) 02 September 2022 (2022-09-02) the whole document	1-23
A	US 2018288446 A1 (MEDIATEK SINGAPORE PTE LTD) 04 October 2018 (2018-10-04) the whole document	1-23
A	US 2021112262 A1 (LG ELECTRONICS INC) 15 April 2021 (2021-04-15) the whole document	1-23

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

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
08 April 2024	**09 April 2024**
Name and mailing address of the ISA/CN **CHINA NATIONAL INTELLECTUAL PROPERTY ADMINISTRATION** **6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing 100088, China**	Authorized officer **YANG,QinZhi** Telephone No. **(+86) 010-62411333**

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2020243246 A1 (BYTEDANCE INC) 03 December 2020 (2020-12-03) the whole document	1-23

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2024/071591

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	113170122	A	23 July 2021	WO	2020108591	A1	04 June 2020
				US	2021344902	A1	04 November 2021
				US	11902507	B2	13 February 2024
CN	113228644	A	06 August 2021	KR	20210104133	A	24 August 2021
				WO	2020140876	A1	09 July 2020
				TW	202037173	A	01 October 2020
				TWI	734328	B	21 July 2021
				EP	3903484	A1	03 November 2021
				EP	3903484	A4	02 November 2022
				US	2022086439	A1	17 March 2022
				US	11930174	B2	12 March 2024
US	2021352304	A1	11 November 2021	EP	3868100	A1	25 August 2021
				EP	3868100	A4	29 December 2021
				KR	20210087088	A	09 July 2021
				KR	102626200	B1	19 January 2024
				KR	20240011253	A	25 January 2024
				CA	3121220	A1	10 September 2020
				US	2022295082	A1	15 September 2022
				US	11716479	B2	01 August 2023
				WO	2020180214	A1	10 September 2020
				BR	112020026183	A2	08 September 2021
				IL	286033	A	31 October 2021
				IL	286033	B	01 November 2022
				IL	286033	B2	01 March 2023
				JP	2022522083	A	14 April 2022
				JP	7251882	B2	04 April 2023
				US	2023353760	A1	02 November 2023
				AU	2020232820	A1	10 June 2021
				AU	2020232820	A8	08 July 2021
				JP	2023085351	A	20 June 2023
				US	11375213	B2	28 June 2022
				MX	2021008404	A	16 August 2021
CN	113892267	A	04 January 2022	WO	2020243246	A1	03 December 2020
CN	115002459	A	02 September 2022	US	2022295109	A1	15 September 2022
				US	11924473	B2	05 March 2024
				WO	2019135628	A1	11 July 2019
				US	2022295112	A1	15 September 2022
				US	11812061	B2	07 November 2023
				US	2022295110	A1	15 September 2022
				KR	20230058343	A	03 May 2023
				KR	20230058033	A	02 May 2023
				KR	20230057318	A	28 April 2023
				US	2022295111	A1	15 September 2022
US	2018288446	A1	04 October 2018	WO	2016074147	A1	19 May 2016
				WO	2016074567	A1	19 May 2016
				EP	4297407	A1	27 December 2023
				US	10390050	B2	20 August 2019
				EP	3205100	A1	16 August 2017
				EP	3205100	A4	16 May 2018

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
				RU	2017114947	A	29 October 2018
				RU	2017114947	A3	06 December 2018
US	2021112262	A1	15 April 2021	WO	2019194496	A1	10 October 2019
				US	11190790	B2	30 November 2021
WO	2020243246	A1	03 December 2020		None		