



(51) International Patent Classification:

H04N 19/52 (2014.01) H04N 19/537 (2014.01)

H04N 19/567 (2014.01)

(21) International Application Number:

PCT/CN2020/089742

(22) International Filing Date:

12 May 2020 (12.05.2020)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

PCT/CN2019/086510

12 May 2019 (12.05.2019)

CN

(71) Applicants: BEIJING BYTEDANCE NETWORK

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

(72) Inventors: ZHANG, Li; 12655 West Jefferson Boulevard,

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

(US). LIU, Hongbin; Jinritoutiao Post Office, China Satellite Communications Tower, No. 63, Zhichun Road, Haidian District, Beijing 100080 (CN). WANG, Yue; Jinritoutiao Post Office, China Satellite Communications Tower, No. 63, Zhichun Road, Haidian District, Beijing 100080 (CN).

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

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

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK,

(54) Title: CODING OF MULTIPLE INTRA PREDICTION METHODS

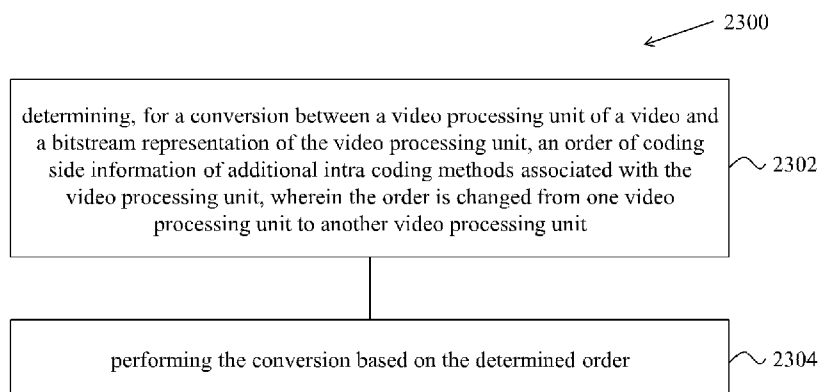


FIG. 23

(57) Abstract: Coding of multiple intra prediction methods is described. In an exemplary aspect, a method for video processing includes determining, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, an order of coding side information of additional intra coding methods associated with the video processing unit, wherein the order is changed from one video processing unit to another video processing unit; and performing the conversion based on the determined order.

EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV,
MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,
TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW,
KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

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

CODING OF MULTIPLE INTRA PREDICTION METHODS

CROSS-REFERENCE TO RELATED APPLICATION

[0001] Under the applicable patent law and/or rules pursuant to the Paris Convention, this application is made to timely claim the priority to and benefits of International Patent Application No. PCT/CN2019/086510, filed on May 12, 2019. The entire disclosures of International Patent Application No. PCT/CN2019/086510 are incorporated by reference as part of the disclosure of this application.

TECHNICAL FIELD

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

BACKGROUND

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

SUMMARY

[0004] Devices, systems and methods related to digital video coding, and specifically, to coding of multiple intra prediction methods in video coding. The described methods may be applied to both the existing video coding standards (e.g., High Efficiency Video Coding (HEVC)) and future video coding standards or video codecs.

[0005] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes configuring an order of coding side information of one or more intra coding methods for a current video block different from a previous order of coding side information for a previous video block, and performing, based on the configuring, a conversion between the current video block and a bitstream representation of the current video block.

[0006] In another representative aspect, the disclosed technology may be used to provide a

method for video processing. This method includes making a decision that a current video block is coded using a coding method selected from a group comprising an affine linear weighted intra prediction (ALWIP) method, a multiple reference line (MRL) intra prediction method, a quantized residual block differential pulse-code modulation (QR-BDPCM) coding method and an intra subblock partitioning (ISP) method, constructing, based on the decision, a most probable mode (MPM) list for the coding method based on a construction process that is common for each coding method in the group, and performing, based on the MPM list, a conversion between the current video block and a bitstream representation of the current video block.

[0007] In yet another representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes configuring, for an application of one or more intra coding methods to a current video block, a bitstream representation of the current video block, wherein at least one of (i) an indication of the application, (ii) syntax elements related to the one or more intra coding methods, (iii) information related to intra-prediction modes or (iv) intra coding information is signaled in the bitstream representation, and performing, based on the configuring, a conversion between the current video block and the bitstream representation of the current video block.

[0008] In yet another representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes configuring, for an application of one or more intra coding methods to a current video block, a bitstream representation of the current video block, wherein a single syntax element that jointly codes the application of the one or more intra coding methods is signaled in the bitstream representation, and performing, based on the configuring, a conversion between the current video block and the bitstream representation of the current video block.

[0009] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes determining, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, an order of coding side information of additional intra coding methods associated with the video processing unit, wherein the order is changed from one video processing unit to another video processing unit; and performing the conversion based on the determined order.

[0010] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes coding, for a conversion between a video

processing unit of the video and a bitstream representation of the video processing unit, an indication of usage of additional intra coding methods or conventional intra prediction method for the video processing unit in the bitstream, wherein the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method, and the conventional intra prediction method includes one or more intra coding methods that use adjacent line or column for intra prediction which uses interpolation filter along prediction direction; and performing the conversion based on the indication.

[0011] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes constructing, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, a most probable mode (MPM) list by unifying a MPM list construction process for all additional intra coding methods associated with the video processing unit; and performing the conversion based on the MPM list.

[0012] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes signaling, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, intra-prediction modes information before usage of one or multiple additional intra coding methods associated with the video processing unit; and performing the conversion based on the intra-prediction modes information.

[0013] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes signaling, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, one or multiple additional intra coding methods related syntax elements after certain intra coding information; and performing the conversion based on the one or multiple additional intra coding methods related syntax elements.

[0014] In one representative aspect, the disclosed technology may be used to provide a method for video processing. This method includes constructing, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, candidate intra coding method list (IPMList) for the video processing unit; and performing the conversion based on the IPMList.

[0015] In one representative aspect, the disclosed technology may be used to provide a

method for video processing. This method includes jointly coding, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, indications of multiple intra coding methods associated with the video processing unit by using one syntax element; and performing the conversion based on the coded indications.

[0016] In yet another representative aspect, the above-described method is embodied in the form of processor-executable code and stored in a computer-readable program medium.

[0017] In yet another representative aspect, a device that is configured or operable to perform the above-described method is disclosed. The device may include a processor that is programmed to implement this method.

[0018] In yet another representative aspect, a video decoder apparatus may implement a method as described herein.

[0019] The above and other aspects and features of the disclosed technology are described in greater detail in the drawings, the description and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] FIG. 1 shows a block diagram of an example encoder.

[0021] FIG. 2 shows an example of 67 intra prediction modes.

[0022] FIG. 3 shows an example of ALWIP for 4×4 blocks.

[0023] FIG. 4 shows an example of ALWIP for 8×8 blocks.

[0024] FIG. 5 shows an example of ALWIP for 8×4 blocks.

[0025] FIG. 6 shows an example of ALWIP for 16×16 blocks.

[0026] FIG. 7 shows an example of four reference lines neighboring a prediction block.

[0027] FIG. 8 shows an example of divisions of 4×8 and 8×4 blocks.

[0028] FIG. 9 shows an example of divisions all blocks except 4×8 , 8×4 and 4×4 .

[0029] FIG. 10 shows an example of a secondary transform in JEM.

[0030] FIG. 11 shows an example of the proposed reduced secondary transform (RST).

[0031] FIG. 12 shows examples of the forward and inverse reduced transforms.

[0032] FIG. 13 shows an example of a forward RST 8×8 process with a 16×48 matrix.

[0033] FIG. 14 shows an example of scanning positions 17 through 64 in an 8×8 block for a non-zero element.

[0034] FIG. 15 shows an example of sub-block transform modes SBT-V and SBT-H.

- [0035] FIG. 16 shows an example of a diagonal up-right scan order for a 4×4 coding group.
- [0036] FIG. 17 shows an example of a diagonal up-right scan order for an 8×8 block with coding groups of size 4×4 .
- [0037] FIG. 18 shows an example of a template used to select probability models.
- [0038] FIG. 19 shows an example of two scalar quantizers used for dependent quantization.
- [0039] FIG. 20 shows an example of a state transition and quantizer selection for the proposed dependent quantization process.
- [0040] FIGS. 21A–21E show flowcharts of example methods for video processing.
- [0041] FIG. 22 is a block diagram of an example of a hardware platform for implementing a visual media decoding or a visual media encoding technique described in the present document.
- [0042] FIG. 23 shows a flowchart of an example method for video processing.
- [0043] FIG. 24 shows a flowchart of an example method for video processing.
- [0044] FIG. 25 shows a flowchart of an example method for video processing.
- [0045] FIG. 26 shows a flowchart of an example method for video processing.
- [0046] FIG. 27 shows a flowchart of an example method for video processing.
- [0047] FIG. 28 shows a flowchart of an example method for video processing.
- [0048] FIG. 29 shows a flowchart of an example method for video processing.

DETAILED DESCRIPTION

[0049] Embodiments of the disclosed technology may be applied to existing video coding standards (e.g., HEVC, H.265) and future standards to improve compression performance. Section headings are used in the present document to improve readability of the description and do not in any way limit the discussion or the embodiments (and/or implementations) to the respective sections only.

2 Video coding introduction

[0050] Due to the increasing demand of higher resolution video, video coding methods and techniques are ubiquitous in modern technology. Video codecs typically include an electronic circuit or software that compresses or decompresses digital video, and are continually being improved to provide higher coding efficiency. A video codec converts uncompressed video to a compressed format or vice versa. There are complex relationships between the video quality, the

amount of data used to represent the video (determined by the bit rate), the complexity of the encoding and decoding algorithms, sensitivity to data losses and errors, ease of editing, random access, and end-to-end delay (latency). The compressed format usually conforms to a standard video compression specification, e.g., the High Efficiency Video Coding (HEVC) standard (also known as H.265 or MPEG-H Part 2), the Versatile Video Coding standard to be finalized, or other current and/or future video coding standards.

[0051] Video coding standards have evolved primarily through the development of the well-known ITU-T and ISO/IEC standards. The ITU-T produced H.261 and H.263, ISO/IEC produced MPEG-1 and MPEG-4 Visual, and the two organizations jointly produced the H.262/MPEG-2 Video and H.264/MPEG-4 Advanced Video Coding (AVC) and H.265/HEVC standards. Since H.262, the video coding standards are based on the hybrid video coding structure wherein temporal prediction plus transform coding are utilized. To explore the future video coding technologies beyond HEVC, Joint Video Exploration Team (JVET) was founded by VCEG and MPEG jointly in 2015. Since then, many new methods have been adopted by JVET and put into the reference software named Joint Exploration Model (JEM). In April 2018, the Joint Video Expert Team (JVET) between VCEG (Q6/16) and ISO/IEC JTC1 SC29/WG11 (MPEG) was created to work on the VVC standard targeting at 50% bitrate reduction compared to HEVC.

2.1 Coding flow of a typical video codec

[0052] FIG. 1 shows an example of encoder block diagram of VVC, which contains three in-loop filtering blocks: deblocking filter (DF), sample adaptive offset (SAO) and ALF. Unlike DF, which uses predefined filters, SAO and ALF utilize the original samples of the current picture to reduce the mean square errors between the original samples and the reconstructed samples by adding an offset and by applying a finite impulse response (FIR) filter, respectively, with coded side information signaling the offsets and filter coefficients. ALF is located at the last processing stage of each picture and can be regarded as a tool trying to catch and fix artifacts created by the previous stages.

2.2 Intra coding in VVC

2.2.1 Intra mode coding with 67 intra prediction modes

[0053] To capture the arbitrary edge directions presented in natural video, the number of directional intra modes is extended from 33, as used in HEVC, to 65. The additional directional

modes are depicted as dotted arrows in FIG. 2, 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.

[0054] Conventional angular intra prediction directions are defined from 45 degrees to -135 degrees in clockwise direction as shown in FIG. 2. In VTM2, several conventional angular intra prediction modes are adaptively replaced with wide-angle intra prediction modes for the non-square blocks. The replaced modes are signaled using the original method and 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 is unchanged.

[0055] In the 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 VVV2, 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.

[0056] In addition to the 67 intra prediction modes, wide-angle intra prediction for non-square blocks (WAIP) and position dependent intra prediction combination (PDPC) methods are further enabled for certain blocks. PDPC is applied to the following intra modes without signalling: planar, DC, horizontal, vertical, bottom-left angular mode and its eight adjacent angular modes, and top-right angular mode and its eight adjacent angular modes.

2.2.2 Affine linear weighted intra prediction (ALWIP or matrix-based intra prediction)

[0057] Affine linear weighted intra prediction (ALWIP, a.k.a. Matrix based intra prediction (MIP)) is proposed.

2.2.2.1 Generation of the reduced prediction signal by matrix vector multiplication

[0058] The neighboring reference samples are firstly down-sampled via averaging to generate the reduced reference signal $bdry_{red}$. Then, the reduced prediction signal $pred_{red}$ is computed by calculating a matrix vector product and adding an offset:

$$[0059] \quad pred_{red} = A \cdot bdry_{red} + b$$

[0060] Here, A is a matrix that has $W_{red} \cdot H_{red}$ rows and 4 columns if $W = H = 4$ and 8 columns in all other cases. b is a vector of size $W_{red} \cdot H_{red}$.

2.2.2.2 Illustration of the entire ALWIP process

[0061] The entire process of averaging, matrix vector multiplication and linear interpolation

is illustrated for different shapes in FIGS. 3–6. Note, that the remaining shapes are treated as in one of the depicted cases.

[0062] 1. Given a 4×4 block, ALWIP takes two averages along each axis of the boundary. The resulting four input samples enter the matrix vector multiplication. The matrices are taken from the set S_0 . After adding an offset, this yields the 16 final prediction samples.

Linear interpolation is not necessary for generating the prediction signal. Thus, a total of $(4 \cdot 16)/(4 \cdot 4) = 4$ multiplications per sample are performed.

[0063] 2. Given an 8×8 block, ALWIP takes four averages along each axis of the boundary. The resulting eight input samples enter the matrix vector multiplication. The matrices are taken from the set S_1 . This yields 16 samples on the odd positions of the prediction block. Thus, a total of $(8 \cdot 16)/(8 \cdot 8) = 2$ multiplications per sample are performed. After adding an offset, these samples are interpolated vertically by using the reduced top boundary. Horizontal interpolation follows by using the original left boundary.

[0064] 3. Given an 8×4 block, ALWIP takes four averages along the horizontal axis of the boundary and the four original boundary values on the left boundary. The resulting eight input samples enter the matrix vector multiplication. The matrices are taken from the set S_1 . This yields 16 samples on the odd horizontal and each vertical positions of the prediction block. Thus, a total of $(8 \cdot 16)/(8 \cdot 4) = 4$ multiplications per sample are performed. After adding an offset, these samples are interpolated horizontally by using the original left boundary.

[0065] 4. Given a 16×16 block, ALWIP takes four averages along each axis of the boundary. The resulting eight input samples enter the matrix vector multiplication. The matrices are taken from the set S_2 . This yields 64 samples on the odd positions of the prediction block. Thus, a total of $(8 \cdot 64)/(16 \cdot 16) = 2$ multiplications per sample are performed. After adding an offset, these samples are interpolated vertically by using eight averages of the top boundary. Horizontal interpolation follows by using the original left boundary. The interpolation process, in this case, does not add any multiplications. Therefore, totally, two multiplications per sample are required to calculate ALWIP prediction.

[0066] For larger shapes, the procedure is essentially the same and it is easy to check that the number of multiplications per sample is less than four.

[0067] For $W \times 8$ blocks with $W > 8$, only horizontal interpolation is necessary as the samples

are given at the odd horizontal and each vertical positions.

[0068] Finally for $W \times 4$ blocks with $W > 8$, let A_kbe the matrix that arises by leaving out every row that corresponds to an odd entry along the horizontal axis of the downsampled block. Thus, the output size is 32 and again, only horizontal interpolation remains to be performed.

[0069] The transposed cases are treated accordingly.

2.2.2.3 Syntax and semantics

7.3.6.5 Coding unit syntax

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
if(tile_group_type != I sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag [x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && tile_group_type != I)	
pred_mode_flag	ae(v)
if(((tile_group_type == I && cu_skip_flag[x0][y0] == 0) (tile_group_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] == MODE_INTRA) {	
if(sps_pcm_enabled_flag && cbWidth >= MinIpcmCbSizeY && cbWidth <= MaxIpcmCbSizeY && cbHeight >= MinIpcmCbSizeY && cbHeight <= MaxIpcmCbSizeY)	
pcm_flag [x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned())	
pcm_alignment_zero_bit	f(1)
pcm_sample(cbWidth, cbHeight, treeType)	
} else {	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_LUMA) {	
if(abs(Log2(cbWidth) - Log2(cbHeight)) <= 2)	
intra_lwip_flag [x0][y0]	ae(v)
if(intra_lwip_flag[x0][y0]) {	
intra_lwip_mpm_flag [x0][y0]	ae(v)
if(intra_lwip_mpm_flag[x0][y0])	
intra_lwip_mpm_idx [x0][y0]	ae(v)
else	
intra_lwip_mpm_remainder [x0][y0]	ae(v)
} else {	
if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx [x0][y0]	ae(v)

if (intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))	
intra_subpartitions_mode_flag [x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_subpartitions_split_flag [x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] == 0 && intra_subpartitions_mode_flag[x0][y0] == 0)	
intra_luma_mpm_flag [x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0])	
intra_luma_mpm_idx [x0][y0]	ae(v)
else	
intra_luma_mpm_remainder [x0][y0]	ae(v)
}	
}	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_CHROMA)	
intra_chroma_pred_mode [x0][y0]	ae(v)
}	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
...	
}	
}	

2.2.3 Multiple reference line (MRL)

[0070] Multiple reference line (MRL) intra prediction uses more reference lines for intra prediction. In FIG. 7, an example of 4 reference lines is depicted, where the samples of segments A and F are not fetched from reconstructed neighbouring samples but padded with the closest samples from Segment B and E, respectively. HEVC intra-picture prediction uses the nearest reference line (i.e., reference line 0). In MRL, 2 additional lines (reference line 1 and reference line 3) are used.

[0071] The index of selected reference line (mrl_idx) is signaled and used to generate intra predictor. For reference line index, which is greater than 0, only include additional reference line modes in MPM list and only signal MPM index without remaining mode. The reference line index is signaled before intra prediction modes, and Planar and DC modes are excluded from intra prediction modes in case a nonzero reference line index is signaled.

[0072] MRL is disabled for the first line of blocks inside a CTU to prevent using extended reference samples outside the current CTU line. Also, PDPC is disabled when additional line is

used.

2.2.4 Intra sub-block partitioning (ISP)

[0073] ISP is proposed, which divides luma intra-predicted blocks vertically or horizontally into 2 or 4 sub-partitions depending on the block size dimensions, as shown in Table 1. FIG. 8 and FIG. 9 show examples of the two possibilities. All sub-partitions fulfill the condition of having at least 16 samples. For block sizes, $4 \times N$ or $N \times 4$ (with $N > 8$), if allowed, the $1 \times N$ or $N \times 1$ sub-partition may exist.

Table 1: Number of sub-partitions depending on the block size (denoted maximum transform size by maxTBSIZE)

Splitting direction	Block Size	Number of Sub-Partitions
N/A	minimum transform size	Not divided
4x8: horizontal 8x4: vertical	4×8 and 8×4	2
Signaled	If neither 4x8 nor 8x4, and $W \leq \text{maxTBSIZE}$ and $H \leq \text{maxTBSIZE}$	4
Horizontal	If not above cases and $H > \text{maxTBSIZE}$	4
Vertical	If not above cases and $W > \text{maxTBSIZE}$	4

[0074] For each of these sub-partitions, a residual signal is generated by entropy decoding the coefficients sent by the encoder and then invert quantizing and invert transforming them. Then, the sub-partition is intra predicted and finally the corresponding reconstructed samples are obtained by adding the residual signal to the prediction signal. Therefore, the reconstructed values of each sub-partition will be available to generate the prediction of the next one, which will repeat the process and so on. All sub-partitions share the same intra mode.

Table 2: Specification of trTypeHor and trTypeVer depending on predModeIntra

predModelIntra	trTypeHor	trTypeVer
INTRA_PLANAR, INTRA_ANGULAR31, INTRA_ANGULAR32, INTRA_ANGULAR34, INTRA_ANGULAR36, INTRA_ANGULAR37	(nTbW >= 4 && nTbW <= 16) ? DST- VII : DCT-II	(nTbH >= 4 && nTbH <= 16) ? DST-VII : DCT-II
INTRA_ANGULAR33, INTRA_ANGULAR35	DCT-II	DCT-II
INTRA_ANGULAR2, INTRA_ANGULAR4,...,INTRA_ANGULAR28, INTRA_ANGULAR30, INTRA_ANGULAR39, INTRA_ANGULAR41,...,INTRA_ANGULAR63, INTRA_ANGULAR65	(nTbW >= 4 && nTbW <= 16) ? DST- VII : DCT-II	DCT-II
INTRA_ANGULAR3, INTRA_ANGULAR5,..., INTRA_ANGULAR27, INTRA_ANGULAR29, INTRA_ANGULAR38, INTRA_ANGULAR40,...,INTRA_ANGULAR64, INTRA_ANGULAR66	DCT-II	(nTbH >= 4 && nTbH <= 16) ? DST-VII : DCT-II

2.2.4.1 Syntax and semantics

7.3.7.5 Coding unit syntax

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
if(slice_type != I sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag [x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && slice_type != I)	
pred_mode_flag	ae(v)
if(((slice_type == I && cu_skip_flag[x0][y0] == 0) (slice_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] == MODE_INTRA) {	
if(sps_pcm_enabled_flag && cbWidth >= MinIpcmCbSizeY && cbWidth <= MaxIpcmCbSizeY && cbHeight >= MinIpcmCbSizeY && cbHeight <= MaxIpcmCbSizeY)	
pcm_flag [x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned())	
pcm_alignment_zero_bit	f(1)
pcm_sample(cbWidth, cbHeight, treeType)	
} else {	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_LUMA) {	
if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx [x0][y0]	ae(v)

if(intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))	
intra_subpartitions_mode_flag [x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_subpartitions_split_flag [x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] == 0 && intra_subpartitions_mode_flag[x0][y0] == 0)	
intra_luma_mpm_flag [x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0])	
intra_luma_mpm_idx [x0][y0]	ae(v)
else	
intra_luma_mpm_remainder [x0][y0]	ae(v)
}	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_CHROMA)	
intra_chroma_pred_mode [x0][y0]	ae(v)
}	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
...	
}	
...	
}	

intra_subpartitions_mode_flag[x0][y0] equal to 1 specifies that the current intra coding unit is partitioned into NumIntraSubPartitions[x0][y0] rectangular transform block subpartitions. **intra_subpartitions_mode_flag**[x0][y0] equal to 0 specifies that the current intra coding unit is not partitioned into rectangular transform block subpartitions.

When **intra_subpartitions_mode_flag**[x0][y0] is not present, it is inferred to be equal to 0.

intra_subpartitions_split_flag[x0][y0] specifies whether the intra subpartitions split type is horizontal or vertical. When **intra_subpartitions_split_flag**[x0][y0] is not present, it is inferred as follows:

- If cbHeight is greater than MaxTbSizeY, **intra_subpartitions_split_flag**[x0][y0] is inferred to be equal to 0.
- Otherwise (cbWidth is greater than MaxTbSizeY), **intra_subpartitions_split_flag**[x0][y0] is inferred to be equal to 1.

The variable IntraSubPartitionsSplitType specifies the type of split used for the current luma coding block as illustrated in Table 7-9. IntraSubPartitionsSplitType is derived as follows:

- If **intra_subpartitions_mode_flag**[x0][y0] is equal to 0, IntraSubPartitionsSplitType is set equal to 0.
- Otherwise, the IntraSubPartitionsSplitType is set equal to 1 + **intra_subpartitions_split_flag**[x0][y0].

Table 7-9 – Name association to IntraSubPartitionsSplitType

IntraSubPartitionsSplitType	Name of IntraSubPartitionsSplitType
0	ISP_NO_SPLIT
1	ISP_HOR_SPLIT
2	ISP_VER_SPLIT

The variable NumIntraSubPartitions specifies the number of transform block subpartitions an intra luma coding block is divided into. NumIntraSubPartitions is derived as follows:

- If IntraSubPartitionsSplitType is equal to ISP_NO_SPLIT, NumIntraSubPartitions is set equal to 1.
- Otherwise, if one of the following conditions is true, NumIntraSubPartitions is set equal to 2:
 - cbWidth is equal to 4 and cbHeight is equal to 8,
 - cbWidth is equal to 8 and cbHeight is equal to 4.
- Otherwise, NumIntraSubPartitions is set equal to 4.

2.3 Transform coding in VVC

2.3.1 Multiple Transform Set (MTS) in VVC

2.3.1.1 Explicit Multiple Transform Set (MTS)

[0075] In VTM4, large block-size transforms, up to 64×64 in size, are enabled, which is primarily useful for higher resolution video, e.g., 1080p and 4K sequences. High frequency transform coefficients are zeroed out for the transform blocks with size (width or height, or both width and height) equal to 64, so that only the lower-frequency coefficients are retained. For example, for an $M \times N$ transform block, with M as the block width and N as the block height, when M is equal to 64, only the left 32 columns of transform coefficients are kept. Similarly, when N is equal to 64, only the top 32 rows of transform coefficients are kept. When transform skip mode is used for a large block, the entire block is used without zeroing out any values.

[0076] In addition to DCT-II which has been employed in HEVC, a Multiple Transform Selection (MTS) scheme is used for residual coding both inter and intra coded blocks. It uses multiple selected transforms from the DCT8/DST7. The newly introduced transform matrices are DST-VII and DCT-VIII. The Table 4 below shows the basis functions of the selected DST/DCT.

Table 4: Basis functions of transform matrices used in VVC

Transform Type	Basis function $T_i(j)$, $i, j=0, 1, \dots, N-1$
DCT-II	$T_i(j) = \omega_0 \cdot \sqrt{\frac{2}{N}} \cdot \cos\left(\frac{\pi \cdot i \cdot (2j + 1)}{2N}\right)$ where $\omega_0 = \begin{cases} \sqrt{\frac{2}{N}} & i = 0 \\ 1 & i \neq 0 \end{cases}$
DCT-VIII	$T_i(j) = \sqrt{\frac{4}{2N + 1}} \cdot \cos\left(\frac{\pi \cdot (2i + 1) \cdot (2j + 1)}{4N + 2}\right)$
DST-VII	$T_i(j) = \sqrt{\frac{4}{2N + 1}} \cdot \sin\left(\frac{\pi \cdot (2i + 1) \cdot (j + 1)}{2N + 1}\right)$

[0077] In order to keep the orthogonality of the transform matrix, the transform matrices are quantized more accurately than the transform matrices in HEVC. To keep the intermediate values of the transformed coefficients within the 16-bit range, after horizontal and after vertical transform, all the coefficients are to have 10-bit.

[0078] In order to control MTS scheme, separate enabling flags are specified at SPS level for intra and inter, respectively. When MTS is enabled at SPS, a CU level flag is signalled to indicate whether MTS is applied or not. Here, MTS is applied only for luma. The MTS CU level flag is signalled when the following conditions are satisfied.

[0079] ○ Both width and height smaller than or equal to 32

[0080] ○ CBF flag is equal to one

[0081] If MTS CU flag is equal to zero, then DCT2 is applied in both directions. However, if MTS CU flag is equal to one, then two other flags are additionally signalled to indicate the transform type for the horizontal and vertical directions, respectively. Transform and signalling mapping table as shown in Table 5. When it comes to transform matrix precision, 8-bit primary transform cores are used. Therefore, all the transform cores used in HEVC are kept as the same, including 4-point DCT-2 and DST-7, 8-point, 16-point and 32-point DCT-2. Also, other transform cores including 64-point DCT-2, 4-point DCT-8, 8-point, 16-point, 32-point DST-7 and DCT-8, use 8-bit primary transform cores.

Table 5: Mapping of decoded value of tu_mts_idx and corresponding transform matrices for the horizontal and vertical directions.

Bin string of tu_mts_idx	tu_mts_idx	Intra/inter	
		Horizontal 1	Vertical
0	0	DCT2	
1 0	1	DST7	DST7
1 1 0	2	DCT8	DST7
1 1 1 0	3	DST7	DCT8
1 1 1 1	4	DCT8	DCT8

[0082] To reduce the complexity of large size DST-7 and DCT-8, High frequency transform coefficients are zeroed out for the DST-7 and DCT-8 blocks with size (width or height, or both width and height) equal to 32. Only the coefficients within the 16x16 lower-frequency region are retained.

[0083] In addition to the cases wherein different transforms are applied, VVC also supports a mode called transform skip (TS) which is like the concept of TS in the HEVC. TS is treated as a special case of MTS.

2.3.2 Reduced Secondary Transform (RST)

2.3.2.1 Non-Separable Secondary Transform (NSST) in JEM

[0084] In JEM, secondary transform is applied between forward primary transform and quantization (at encoder) and between de-quantization and invert primary transform (at decoder side). As shown in FIG. 10, 4x4 (or 8x8) secondary transform is performed depends on block size. For example, 4x4 secondary transform is applied for small blocks (i.e., min (width, height) < 8) and 8x8 secondary transform is applied for larger blocks (i.e., min (width, height) > 4) per 8x8 block.

[0085] Application of a non-separable transform is described as follows using input as an example. To apply the non-separable transform, the 4x4 input block X

$$\text{[0086]} \quad X = \begin{bmatrix} X_{00} & X_{01} & X_{02} & X_{03} \\ X_{10} & X_{11} & X_{12} & X_{13} \\ X_{20} & X_{21} & X_{22} & X_{23} \\ X_{30} & X_{31} & X_{32} & X_{33} \end{bmatrix}$$

[0087] is first represented as a vector $\vec{X}$:

$$\text{[0088]} \quad \vec{X} =$$

$$[X_{00} \ X_{01} \ X_{02} \ X_{03} \ X_{10} \ X_{11} \ X_{12} \ X_{13} \ X_{20} \ X_{21} \ X_{22} \ X_{23} \ X_{30} \ X_{31} \ X_{32} \ X_{33}]^T$$

[0089] The non-separable transform is calculated as $\vec{F} = T \cdot \vec{X}$, where $\vec{F}$ indicates the transform coefficient vector, and T is a 16x16 transform matrix. The 16x1 coefficient vector $\vec{F}$ is subsequently re-organized as 4x4 block using the scanning order for that block (horizontal, vertical or diagonal). The coefficients with smaller index will be placed with the smaller scanning index in the 4x4 coefficient block. There are totally 35 transform sets and 3 non-separable transform matrices (kernels) per transform set are used. The mapping from the intra prediction mode to the transform set is pre-defined. For each transform set, the selected non-separable secondary transform (NSST) candidate is further specified by the explicitly signalled secondary transform index. The index is signalled in a bit-stream once per Intra CU after transform coefficients.

2.3.2.2 Reduced Secondary Transform (RST)

[0090] The RST (a.k.a. Low Frequency Non-Separable Transform (LFNST)) was introduced and 4 transform set (instead of 35 transform sets) mapping was introduced. 16x64 (further reduced to 16x48) and 16x16 matrices are employed. For notational convenience, the 16x64 (reduced to 16x48) transform is denoted as RST8x8 and the 16x16 one as RST4x4. FIG. 11 shows an example of RST.

2.3.2.2.1 RST computation

[0091] The main idea of a Reduced Transform (RT) is to map an N dimensional vector to an R dimensional vector in a different space, where R/N ($R < N$) is the reduction factor.

[0092] The RT matrix is an $R \times N$ matrix as follows:

$$T_{R \times N} = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \cdots & t_{1N} \\ t_{21} & t_{22} & t_{23} & \cdots & t_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{R1} & t_{R2} & t_{R3} & \cdots & t_{RN} \end{bmatrix}$$

[0093] where the R rows of the transform are R bases of the N dimensional space. The invert transform matrix for RT is the transpose of its forward transform. The forward and invert RT are depicted in FIG. 12.

[0094] In this contribution, the RST8x8 with a reduction factor of 4 (1/4 size) is applied. Hence, instead of 64x64, which is conventional 8x8 non-separable transform matrix size, 16x64 direct matrix is used. In other words, the 64x16 invert RST matrix is used at the decoder side to

generate core (primary) transform coefficients in 8×8 top-left regions. The forward RST 8×8 uses 16×64 (or 8×64 for 8×8 block) matrices so that it produces non-zero coefficients only in the top-left 4×4 region within the given 8×8 region. In other words, if RST is applied then the 8×8 region except the top-left 4×4 region will have only zero coefficients. For RST 4×4 , 16×16 (or 8×16 for 4×4 block) direct matrix multiplication is applied.

[0095] An invert RST is conditionally applied when the following two conditions are satisfied:

[0096] ○ Block size is greater than or equal to the given threshold ($W \geq 4 \ \&\& \ H \geq 4$)

[0097] ○ Transform skip mode flag is equal to zero

[0098] If both width (W) and height (H) of a transform coefficient block is greater than 4, then the RST 8×8 is applied to the top-left 8×8 region of the transform coefficient block. Otherwise, the RST 4×4 is applied on the top-left $\min(8, W) \times \min(8, H)$ region of the transform coefficient block.

[0099] If RST index is equal to 0, RST is not applied. Otherwise, RST is applied, of which kernel is chosen with the RST index. The RST selection method and coding of the RST index are explained later.

[00100] Furthermore, RST is applied for intra CU in both intra and inter slices, and for both Luma and Chroma. If a dual tree is enabled, RST indices for Luma and Chroma are signaled separately. For inter slice (the dual tree is disabled), a single RST index is signaled and used for both Luma and Chroma.

2.3.2.2.2 Restriction of RST

[00101] When ISP mode is selected, RST is disabled, and RST index is not signaled, because performance improvement was marginal even if RST is applied to every feasible partition block. Furthermore, disabling RST for ISP-predicted residual could reduce encoding complexity.

2.3.2.2.3 RST selection

[00102] A RST matrix is chosen from four transform sets, each of which consists of two transforms. Which transform set is applied is determined from intra prediction mode as the following:

[00103] (1) If one of three CCLM modes is indicated, transform set 0 is selected.

[00104] (2) Otherwise, transform set selection is performed according to the following table:

The transform set selection table

IntraPredMode	Tr. set index
IntraPredMode < 0	1
0 <= IntraPredMode <= 1	0
2 <= IntraPredMode <= 12	1
13 <= IntraPredMode <= 23	2
24 <= IntraPredMode <= 44	3
45 <= IntraPredMode <= 55	2
56 <= IntraPredMode	1

[00105] The index to access the above table, denoted as IntraPredMode, have a range of [-14, 83], which is a transformed mode index used for wide angle intra prediction.

2.3.2.2.4 RST matrices of reduced dimension

[00106] As a further simplification, 16x48 matrices are applied instead of 16x64 with the same transform set configuration, each of which takes 48 input data from three 4x4 blocks in a top-left 8x8 block excluding right-bottom 4x4 block (as shown in FIG. 13).

2.3.2.2.5 RST signaling

[00107] The forward RST8x8 with R=16 uses 16x64 matrices so that it produces non-zero coefficients only in the top-left 4x4 region within the given 8x8 region. In other words, if RST is applied then the 8x8 region except the top-left 4x4 region generates only zero coefficients. As a result, RST index is not coded when any non-zero element is detected within 8x8 block region other than top-left 4x4 (which is depicted in FIG. 14) because it implies that RST was not applied. In such a case, RST index is inferred to be zero.

2.3.2.2.6 Zero-out region within one CG

[00108] Usually, before applying the invert RST on a 4x4 sub-block, any coefficient in the 4x4 sub-block may be non-zero. However, it is constrained that in some cases, some coefficients in the 4x4 sub-block must be zero before invert RST is applied on the sub-block.

[00109] Let nonZeroSize be a variable. It is required that any coefficient with the index no smaller than nonZeroSize when it is rearranged into a 1-D array before the invert RST must be zero.

[00110] When nonZeroSize is equal to 16, there is no zero-out constrain on the coefficients in the top-left 4x4 sub-block.

[00111] When the current block size is 4x4 or 8x8, nonZeroSize is set equal to 8 (that is,

coefficients with the scanning index in the range [8, 15] as show in FIG. 14, shall be 0). For other block dimensions, nonZeroSize is set equal to 16.

2.3.2.2.7 Description of RST in working Draft

7.3.2.3 Sequence parameter set RBSP syntax

seq_parameter_set_rbsp() {	Descriptor
.....	
sps_mts_enabled_flag	u(1)
if(sps_mts_enabled_flag) {	
sps_explicit_mts_intra_enabled_flag	u(1)
sps_explicit_mts_inter_enabled_flag	u(1)
}	
...	
sps_st_enabled_flag	u(1)
...	
}	

7.3.7.11 Residual coding syntax

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cldx) {	Descriptor
...	
if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDeSigCoeffFlag) && (xC != LastSignificantCoefX yC != LastSignificantCoefY)) {	
sig_coeff_flag [xC][yC]	ae(v)
remBinsPass1--	
if(sig_coeff_flag[xC][yC])	
inferSbDeSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
if(!transform_skip_flag[x0][y0]) {	
numSigCoeff++	
if(((log2TbWidth == 2 && log2TbHeight == 2) (log2TbWidth == 3 && log2TbHeight == 3)) && n >= 8 && i == 0) ((log2TbWidth >= 3 && log2TbHeight >= 3 && (i == 1 i == 2)))) {	
numZeroOutSigCoeff++	
}	
}	
}	
abs_level_gt1_flag [n]	ae(v)
...	

7.3.7.5 Coding unit syntax

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
...	
if(!pcm_flag[x0][y0]) {	

if(CuPredMode[x0][y0] != MODE_INTRA && merge_flag[x0][y0] == 0)	
cu_cbf	ae(v)
if(cu_cbf) {	
if(CuPredMode[x0][y0] == MODE_INTER && sps_sbt_enabled_flag && !ciip_flag[x0][y0]) {	
if(cbWidth <= MaxSbtSize && cbHeight <= MaxSbtSize) {	
allowSbtVerH = cbWidth >= 8	
allowSbtVerQ = cbWidth >= 16	
allowSbtHorH = cbHeight >= 8	
allowSbtHorQ = cbHeight >= 16	
if(allowSbtVerH allowSbtHorH allowSbtVerQ allowSbtHorQ)	
cu_sbt_flag	ae(v)
}	
if(cu_sbt_flag) {	
if((allowSbtVerH allowSbtHorH) && (allowSbtVerQ allowSbtHorQ))	
cu_sbt_quad_flag	ae(v)
if((cu_sbt_quad_flag && allowSbtVerQ && allowSbtHorQ) (cu_sbt_quad_flag && allowSbtVerH && allowSbtHorH))	
cu_sbt_horizontal_flag	ae(v)
cu_sbt_pos_flag	ae(v)
}	
}	
numZeroOutSigCoeff = 0	
transform_tree(x0, y0, cbWidth, cbHeight, treeType)	
if(Min(cbWidth, cbHeight) >= 4 && sps_st_enabled_flag == 1 && CuPredMode[x0][y0] == MODE_INTRA && IntraSubPartitionsSplitType == ISP_NO_SPLIT) {	
if((numSigCoeff > ((treeType == SINGLE_TREE) ? 2 : 1)) && numZeroOutSigCoeff == 0) {	
st_idx[x0][y0]	ae(v)
}	
}	
}	
}	
}	

sps_st_enabled_flag equal to 1 specifies that **st_idx** may be present in the residual coding syntax for intra coding units. **sps_st_enabled_flag** equal to 0 specifies that **st_idx** is not present in the residual coding syntax for intra coding units.

st_idx[x0][y0] specifies which secondary transform kernel is applied between two candidate kernels in a selected transform set. **st_idx[x0][y0]** equal to 0 specifies that the secondary transform is not applied. The array indices **x0**, **y0** specify the location (**x0**, **y0**) of the top-left sample of the considered transform block relative to the top-left sample of the picture.

When **st_idx[x0][y0]** is not present, **st_idx[x0][y0]** is inferred to be equal to 0.

2.3.3 Sub-block transform

[00112] For an inter-predicted CU with `cu_cbf` equal to 1, `cu_sbt_flag` may be signaled to indicate whether the whole residual block or a sub-part of the residual block is decoded. In the former case, inter MTS information is further parsed to determine the transform type of the CU. In the latter case, a part of the residual block is coded with inferred adaptive transform and the other part of the residual block is zeroed out. The SBT is not applied to the combined inter-intra mode.

[00113] In sub-block transform, position-dependent transform is applied on luma transform blocks in SBT-V and SBT-H (chroma TB always using DCT-2). The two positions of SBT-H and SBT-V are associated with different core transforms. More specifically, the horizontal and vertical transforms for each SBT position is specified in Fig. 15. For example, the horizontal and vertical transforms for SBT-V position 0 is DCT-8 and DST-7, respectively. When one side of the residual TU is greater than 32, the corresponding transform is set as DCT-2. Therefore, the sub-block transform jointly specifies the TU tiling, cbf, and horizontal and vertical transforms of a residual block, which may be considered a syntax shortcut for the cases that the major residual of a block is at one side of the block.

2.3.3.1 Syntax elements

7.3.7.5 Coding unit syntax

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
if(slice_type != I sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag [x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && slice_type != I)	
pred_mode_flag	ae(v)
if(((slice_type == I && cu_skip_flag[x0][y0] == 0) (slice_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] == MODE_INTRA) {	
...	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
...	
}	
if(!pcm_flag[x0][y0]) {	
if(CuPredMode[x0][y0] != MODE_INTRA && merge_flag[x0][y0] == 0)	
cu_cbf	ae(v)

if(cu_cbf) {	
if(CuPredMode[x0][y0] == MODE_INTER && sps_sbt_enabled_flag && !ciip_flag[x0][y0]) {	
if(cbWidth <= MaxSbtSize && cbHeight <= MaxSbtSize) {	
allowSbtVerH = cbWidth >= 8	
allowSbtVerQ = cbWidth >= 16	
allowSbtHorH = cbHeight >= 8	
allowSbtHorQ = cbHeight >= 16	
if(allowSbtVerH allowSbtHorH allowSbtVerQ allowSbtHorQ)	
cu_sbt_flag	ae(v)
}	
if(cu_sbt_flag) {	
if((allowSbtVerH allowSbtHorH) && (allowSbtVerQ allowSbtHorQ))	
cu_sbt_quad_flag	ae(v)
if((cu_sbt_quad_flag && allowSbtVerQ && allowSbtHorQ) (!cu_sbt_quad_flag && allowSbtVerH && allowSbtHorH))	
cu_sbt_horizontal_flag	ae(v)
cu_sbt_pos_flag	ae(v)
}	
transform_tree(x0, y0, cbWidth, cbHeight, treeType)	
}	
}	
}	

cu_sbt_flag equal to 1 specifies that for the current coding unit, subblock transform is used. **cu_sbt_flag** equal to 0 specifies that for the current coding unit, subblock transform is not used.

When **cu_sbt_flag** is not present, its value is inferred to be equal to 0.

NOTE – : When subblock transform is used, a coding unit is split into two transform units; one transform unit has residual data, the other does not have residual data.

cu_sbt_quad_flag equal to 1 specifies that for the current coding unit, the subblock transform includes a transform unit of 1/4 size of the current coding unit. **cu_sbt_quad_flag** equal to 0 specifies that for the current coding unit the subblock transform includes a transform unit of 1/2 size of the current coding unit.

When **cu_sbt_quad_flag** is not present, its value is inferred to be equal to 0.

cu_sbt_horizontal_flag equal to 1 specifies that the current coding unit is split horizontally into 2 transform units. **cu_sbt_horizontal_flag**[x0][y0] equal to 0 specifies that the current coding unit is split vertically into 2 transform units.

When **cu_sbt_horizontal_flag** is not present, its value is derived as follows:

- If **cu_sbt_quad_flag** is equal to 1, **cu_sbt_horizontal_flag** is set to be equal to **allowSbtHorQ**.
- Otherwise (**cu_sbt_quad_flag** is equal to 0), **cu_sbt_horizontal_flag** is set to be equal to **allowSbtHorH**.

cu_sbt_pos_flag equal to 1 specifies that the **tu_cbf_luma**, **tu_cbf_cb** and **tu_cbf_cr** of the first transform unit in the current coding unit are not present in the bitstream. **cu_sbt_pos_flag** equal to 0 specifies that the **tu_cbf_luma**, **tu_cbf_cb** and **tu_cbf_cr** of the second transform unit in the current coding unit are not present in the bitstream.

The variable **SbtNumFourthsTb0** is derived as follows:

$$\text{sbtMinNumFourths} = \text{cu_sbt_quad_flag} ? 1 : 2 \quad (7-117)$$

$$\text{SbtNumFourthsTb0} = \text{cu_sbt_pos_flag} ? (4 - \text{sbtMinNumFourths}) : \text{sbtMinNumFourths} \quad (7-118)$$

sps_sbt_max_size_64_flag equal to 0 specifies that the maximum CU width and height for allowing subblock transform is 32 luma samples. **sps_sbt_max_size_64_flag** equal to 1 specifies that the maximum CU width and height for allowing subblock transform is 64 luma samples.

$$\text{MaxSbtSize} = \text{sps_sbt_max_size_64_flag} ? 64 : 32 \quad (7-33)$$

2.3.4 Quantized residual domain Block differential pulse-code modulation coding (QR-BDPCM)

[00114] Quantized residual domain BDPCM (denote as RBDPCM hereinafter) is proposed. The intra prediction is done on the entire block by sample copying in prediction direction (horizontal or vertical prediction) similar to intra prediction. The residual is quantized and the delta between the quantized residual and its predictor (horizontal or vertical) quantized value is coded.

[00115] For a block of size M (rows) $\times$ N (cols), let $r_{i,j}$, $0 \leq i \leq M - 1$, $0 \leq j \leq N - 1$, be the prediction residual after performing intra prediction horizontally (copying left neighbor pixel value across the the predicted block line by line) or vertically (copying top neighbor line to each line in the predicted block) using unfiltered samples from above or left block boundary samples. Let $Q(r_{i,j})$, $0 \leq i \leq M - 1$, $0 \leq j \leq N - 1$ denote the quantized version of the residual $r_{i,j}$, where residual is difference between original block and the predicted block values. Then the block DPCM is applied to the quantized residual samples, resulting in modified $M \times N$ array $\tilde{R}$ with elements $\tilde{r}_{i,j}$. When vertical BDPCM is signaled:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, \quad 0 \leq j \leq (N - 1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \leq i \leq (M - 1), \quad 0 \leq j \leq (N - 1) \end{cases}$$

[00117] For horizontal prediction, similar rules apply, and the residual quantized samples are obtained by

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \leq i \leq (M - 1), \quad j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \leq i \leq (M - 1), \quad 1 \leq j \leq (N - 1) \end{cases}$$

[00119] The residual quantized samples $\tilde{r}_{i,j}$ are sent to the decoder.

[00120] On the decoder side, the above calculations are reversed to produce $Q(r_{i,j})$, $0 \leq i \leq M - 1$, $0 \leq j \leq N - 1$. For vertical prediction case,

$$Q(r_{i,j}) = \sum_{k=0}^i \tilde{r}_{k,j}, \quad 0 \leq i \leq (M - 1), \quad 0 \leq j \leq (N - 1)$$

[00122] For horizontal case,

[00123] $Q(r_{i,j}) = \sum_{k=0}^j \tilde{r}_{i,k}, \quad 0 \leq i \leq (M-1), \quad 0 \leq j \leq (N-1)$

[00124] The invert quantized residuals, $Q^{-1}(Q(r_{i,j}))$, are added to the intra block prediction values to produce the reconstructed sample values.

[00125] When QR-BDPCM is selected, there is no transform applied.

2.4 Entropy coding of coefficients

2.4.1 Coefficients coding of transform-applied blocks

[00126] In HEVC, transform coefficients of a coding block are coded using non-overlapped coefficient groups (or subblocks), and each CG contains the coefficients of a 4x4 block of a coding block. The CGs inside a coding block, and the transform coefficients within a CG, are coded according to pre-defined scan orders.

[00127] The CGs inside a coding block, and the transform coefficients within a CG, are coded according to pre-defined scan orders. Both CG and coefficients within a CG follows the diagonal up-right scan order. An example for 4x4 block and 8x8 scanning order is depicted in FIG. 16 and FIG. 17, respectively.

[00128] Note that the coding order is the reversed scanning order (i.e., decoding from CG3 to CG0 in FIG. 17), when decoding one block, the last non-zero coefficient's coordinate is firstly decoded.

[00129] The coding of transform coefficient levels of a CG with at least one non-zero transform coefficient may be separated into multiple scan passes. In the first pass, the first bin (denoted by bin0, also referred as significant_coeff_flag, which indicates the magnitude of the coefficient is larger than 0) is coded. Next, two scan passes for context coding the second/third bins (denoted by bin1 and bin2, respectively, also referred as coeff_abs_greater1_flag and coeff_abs_greater2_flag) may be applied. Finally, two more scan passes for coding the sign information and the remaining values (also referred as coeff_abs_level_remaining) of coefficient levels are invoked, if necessary. Note that only bins in the first three scan passes are coded in a regular mode and those bins are termed regular bins in the following descriptions.

[00130] In the VVC 3, for each CG, the regular coded bins and the bypass coded bins are separated in coding order; first all regular coded bins for a subblock are transmitted and, thereafter, the bypass coded bins are transmitted. The transform coefficient levels of a subblock are coded in five passes over the scan positions as follows:

[00131] ○ **Pass 1:** coding of significance (sig_flag), greater 1 flag (gt1_flag), parity (par_level_flag) and greater 2 flags (gt2_flag) is processed in coding order. If sig_flag is equal to 1, first the gt1_flag is coded (which specifies whether the absolute level is greater than 1). If gt1_flag is equal to 1, the par_flag is additionally coded (it specifies the parity of the absolute level minus 2).

[00132] ○ **Pass 2:** coding of remaining absolute level (remainder) is processed for all scan positions with gt2_flag equal to 1 or gt1_flag equal to 1. The non-binary syntax element is binarized with Golomb-Rice code and the resulting bins are coded in the bypass mode of the arithmetic coding engine.

[00133] ○ **Pass 3:** absolute level (absLevel) of the coefficients for which no sig_flag is coded in the first pass (due to reaching the limit of regular-coded bins) are completely coded in the bypass mode of the arithmetic coding engine using a Golomb-Rice code.

[00134] ○ **Pass 4:** coding of the signs (sign_flag) for all scan positions with sig_coeff_flag equal to 1

[00135] It is guaranteed that no more than 32 regular-coded bins (sig_flag, par_flag, gt1_flag and gt2_flag) are encoded or decoded for a 4x4 subblock. For 2x2 chroma subblocks, the number of regular-coded bins is limited to 8.

[00136] The Rice parameter (ricePar) for coding the non-binary syntax element remainder (in Pass 3) is derived similar to HEVC. At the start of each subblock, ricePar is set equal to 0. After coding a syntax element remainder, the Rice parameter is modified according to predefined equation. For coding the non-binary syntax element absLevel (in Pass 4), the sum of absolute values sumAbs in a local template is determined. The variables ricePar and posZero are determined based on dependent quantization and sumAbs by a table look-up. The intermediate variable codeValue is derived as follows:

[00137] ○ If absLevel[k] is equal to 0, codeValue is set equal to posZero;

[00138] ○ Otherwise, if absLevel[k] is less than or equal to posZero, codeValue is set equal to absLevel[k] – 1;

[00139] ○ Otherwise (absLevel[k] is greater than posZero), codeValue is set equal to absLevel[k].

[00140] The value of codeValue is coded using a Golomb-Rice code with Rice parameter ricePar.

2.4.1.1 Context modeling for coefficient coding

[00141] The selection of probability models for the syntax elements related to absolute values of transform coefficient levels depends on the values of the absolute levels or partially reconstructed absolute levels in a local neighbourhood. The template used is illustrated in FIG. 18.

[00142] The selected probability models depend on the sum of the absolute levels (or partially reconstructed absolute levels) in a local neighborhood and the number of absolute levels greater than 0 (given by the number of sig_coeff_flags equal to 1) in the local neighborhood. The context modelling and binarization depends on the following measures for the local neighborhood:

[00143] ○ numSig: the number of non-zero levels in the local neighborhood;

[00144] ○ sumAbs1: the sum of partially reconstructed absolute levels (absLevel1) after the first pass in the local neighborhood;

[00145] ○ sumAbs: the sum of reconstructed absolute levels in the local neighborhood

[00146] ○ diagonal position (d): the sum of the horizontal and vertical coordinates of a current scan position inside the transform block

[00147] Based on the values of numSig, sumAbs1, and d, the probability models for coding sig_flag, par_flag, gt1_flag, and gt2_flag are selected. The Rice parameter for binarizing abs_remainder is selected based on the values of sumAbs and numSig.

2.4.1.2 Dependent Quantization (DQ)

[00148] In addition, the same HEVC scalar quantization is used with a new concept called dependent scale quantization. Dependent scalar quantization refers to an approach in which the set of admissible reconstruction values for a transform coefficient depends on the values of the transform coefficient levels that precede the current transform coefficient level in reconstruction order. The main effect of this approach is that, in comparison to conventional independent scalar quantization as used in HEVC, the admissible reconstruction vectors are packed denser in the N-dimensional vector space (N represents the number of transform coefficients in a transform block). That means, for a given average number of admissible reconstruction vectors per N-dimensional unit volume, the average distortion between an input vector and the closest reconstruction vector is reduced. The approach of dependent scalar quantization is realized by: (a) defining two scalar quantizers with different reconstruction levels and (b) defining a process

for switching between the two scalar quantizers.

[00149] The two scalar quantizers used, denoted by Q0 and Q1, are illustrated in FIG. 19. The location of the available reconstruction levels is uniquely specified by a quantization step size Δ . The scalar quantizer used (Q0 or Q1) is not explicitly signalled in the bitstream. Instead, the quantizer used for a current transform coefficient is determined by the parities of the transform coefficient levels that precede the current transform coefficient in coding/reconstruction order.

[00150] As illustrated in FIG. 20, the switching between the two scalar quantizers (Q0 and Q1) is realized via a state machine with four states. The state can take four different values: 0, 1, 2, 3. It is uniquely determined by the parities of the transform coefficient levels preceding the current transform coefficient in coding/reconstruction order. At the start of the inverse quantization for a transform block, the state is set equal to 0. The transform coefficients are reconstructed in scanning order (i.e., in the same order they are entropy decoded). After a current transform coefficient is reconstructed, the state is updated as shown in FIG. 20, where k denotes the value of the transform coefficient level.

2.4.1.3 Syntax and semantics

7.3.7.11 Residual coding syntax

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if((tu_mts_idx[x0][y0] > 0 (cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6) && cIdx == 0 && log2TbWidth > 4)	
log2TbWidth = 4	
else	
log2TbWidth = Min(log2TbWidth, 5)	
if(tu_mts_idx[x0][y0] > 0 (cu_sbt_flag && log2TbWidth < 6 && log2TbHeight < 6) && cIdx == 0 && log2TbHeight > 4)	
log2TbHeight = 4	
else	
log2TbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	

log2SbH = log2SbW	
if (log2TbWidth < 2 && cIdx == 0) {	
log2SbW = log2TbWidth	
log2SbH = 4 - log2SbW	
} else if (log2TbHeight < 2 && cIdx == 0) {	
log2SbH = log2TbHeight	
log2SbW = 4 - log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight - (log2SbW + log2SbH))) - 1	
do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock--	
}	
lastScanPos--	
xS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH]	
[lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH]	
[lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) (yC != LastSignificantCoeffY))	
QState = 0	
for(i = lastSubBlock; i >= 0; i--) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH]	
[lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH]	
[lastSubBlock][1]	
inferSbDcSigCoeffFlag = 0	
if((i < lastSubBlock) && (i > 0)) {	
coded_sub_block_flag [xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = -1	
remBinsPass1 = ((log2SbW + log2SbH) < 4 ? 8 : 32)	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff - 1)	
firstPosMode1 = -1	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n--) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	

yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX yC != LastSignificantCoeffY)) {	
sig_coeff_flag [xC][yC]	ae(v)
remBinsPass1--	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gt1_flag [n]	ae(v)
remBinsPass1--	
if(abs_level_gt1_flag[n]) {	
par_level_flag [n]	ae(v)
remBinsPass1--	
abs_level_gt3_flag [n]	ae(v)
remBinsPass1--	
}	
if(lastSigScanPosSb == -1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gt1_flag[n] + 2 * abs_level_gt3_flag[n]	
if(dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
if(remBinsPass1 < 4)	
firstPosModel = n - 1	
}	
for(n = numSbCoeff - 1; n >= firstPosModel; n--) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(abs_level_gt3_flag[n])	
abs_remainder [n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] + 2 * abs_remainder[n]	
}	
for(n = firstPosModel; n >= 0; n--) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
dec_abs_level [n]	ae(v)
if(AbsLevel[xC][yC] > 0)	
firstSigScanPosSb = n	
if(dep_quant_enabled_flag)	

QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(dep_quant_enabled_flag !sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb - firstSigScanPosSb > 3 ? 1 : 0)	
for(n = numSbCoeff - 1; n >= 0; n--) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHidden (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	
if(dep_quant_enabled_flag) {	
QState = startQStateSb	
for(n = numSbCoeff - 1; n >= 0; n--) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0)	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] - (QState > 1 ? 1 : 0)) * (1 - 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][par_level_flag[n]]	
} else {	
sumAbsLevel = 0	
for(n = numSbCoeff - 1; n >= 0; n--) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 - 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = -TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
}	

2.4.2 Coefficients coding of TS-coded blocks and QR-BDPCM coded blocks

[00151] QR-BDPCM follows the context modeling method for TS-coded blocks.

[00152] A modified transform coefficient level coding for the TS residual. Relative to the regular residual coding case, the residual coding for TS includes the following changes:

[00153] (1) no signaling of the last x/y position

[00154] (2) coded_sub_block_flag coded for every subblock except for the last subblock when all previous flags are equal to 0;

[00155] (3) sig_coeff_flag context modelling with reduced template,

[00156] (4) a single context model for abs_level_gt1_flag and par_level_flag,

[00157] (5) context modeling for the sign flag, additional greater than 5, 7, 9 flags,

[00158] (6) modified Rice parameter derivation for the remainder binarization

[00159] (7) a limit for the number of context coded bins per sample, 2 bins per sample within one block.

2.4.2.1 Syntax and semantics

7.3.6.10 Transform unit syntax

transform_unit(x0, y0, tbWidth, tbHeight, treeType, subTuIndex) {	Descriptor
...	
if(tu_cbf_luma[x0][y0] && treeType != DUAL_TREE_CHROMA && (tbWidth <= 32) && (tbHeight <= 32) && (IntraSubPartitionsSplit[x0][y0] == ISP_NO_SPLIT) && (!cu_sbt_flag)) {	
if(transform_skip_enabled_flag && tbWidth <= MaxTsSize && tbHeight <= MaxTsSize)	
transform_skip_flag[x0][y0]	ae(v)
if(((CuPredModel[x0][y0] != MODE_INTRA && sps_explicit_mts_inter_enabled_flag) (CuPredModel[x0][y0] == MODE_INTRA && sps_explicit_mts_intra_enabled_flag)) && (tbWidth <= 32) && (tbHeight <= 32) && (!transform_skip_flag[x0][y0]))	
tu_mts_idx[x0][y0]	ae(v)
}	
if(tu_cbf_luma[x0][y0]) {	
if(!transform_skip_flag[x0][y0])	
residual_coding(x0, y0, Log2(tbWidth), Log2(tbHeight), 0)	
else	
residual_coding_ts(x0, y0, Log2(tbWidth), Log2(tbHeight), 0)	
}	
if(tu_cbf_cb[x0][y0])	
residual_coding(xC, yC, Log2(wC), Log2(hC), 1)	
if(tu_cbf_cr[x0][y0])	
residual_coding(xC, yC, Log2(wC), Log2(hC), 2)	
}	

	Descriptor
residual_ts_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	
log2SbSize = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
numSbCoeff = 1 << (log2SbSize << 1)	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight - 2 * log2SbSize)) - 1	
/* Loop over subblocks from top-left (DC) subblock to the last one */	
inferSbCbf = 1	
MaxCcbs = 2 * (1 << log2TbWidth) * (1 << log2TbHeight)	
for(i = 0; i <= lastSubBlock; i++) {	
xS = DiagScanOrder[log2TbWidth - log2SbSize][log2TbHeight - log2SbSize][i][0]	
yS = DiagScanOrder[log2TbWidth - log2SbSize][log2TbHeight - log2SbSize][i][1]	
if((i != lastSubBlock !inferSbCbf)	
coded_sub_block_flag [xS][yS]	ae(v)
MaxCcbs--	
if(coded_sub_block_flag[xS][yS] && i < lastSubBlock)	
inferSbCbf = 0	
}	
/* First scan pass */	
inferSbSigCoeffFlag = 1	
for(n = (i == 0; n <= numSbCoeff - 1; n++) {	
xC = (xS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) + DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(coded_sub_block_flag[xS][yS] &&	
(n == numSbCoeff - 1 !inferSbSigCoeffFlag)) {	
sig_coeff_flag [xC][yC]	ae(v)
MaxCcbs--	
if(sig_coeff_flag[xC][yC])	
inferSbSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
coeff_sign_flag [n]	ae(v)
abs_level_gtx_flag [n][0]	ae(v)
MaxCcbs = MaxCcbs - 2	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag [n]	ae(v)
MaxCcbs--	
}	
}	
AbsLevelPassX[xC][yC] =	
sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0]	
}	
/* Greater than X scan passes (numGtXFlags=5) */	
for(i = 1; i <= 5 - 1 && abs_level_gtx_flag[n][i - 1]; i++) {	
for(n = 0; n <= numSbCoeff - 1; n++) {	

$x_C = (x_S \ll \log_2 \text{SbSize}) + \text{DiagScanOrder}[\log_2 \text{SbSize}][\log_2 \text{SbSize}][n][0]$	
$y_C = (y_S \ll \log_2 \text{SbSize}) + \text{DiagScanOrder}[\log_2 \text{SbSize}][\log_2 \text{SbSize}][n][1]$	
abs_level_gtx_flag [n][i]	ae(v)
MaxCcbS --	
$\text{AbsLevelPassX}[x_C][y_C] += 2 * \text{abs_level_gtx_flag}[n][i]$	
}	
}	
/* remainder scan pass */	
for(n = 0; n <= numSbCoeff - 1; n++) {	
$x_C = (x_S \ll \log_2 \text{SbSize}) + \text{DiagScanOrder}[\log_2 \text{SbSize}][\log_2 \text{SbSize}][n][0]$	
$y_C = (y_S \ll \log_2 \text{SbSize}) + \text{DiagScanOrder}[\log_2 \text{SbSize}][\log_2 \text{SbSize}][n][1]$	
if(abs_level_gtx_flag[n][numGtXFlags - 1])	
abs_remainder [n]	ae(v)
$\text{TransCoeffLevel}[x_0][y_0][cIdx][x_C][y_C] = (1 - 2 * \text{coeff_sign_flag}[n]) * (\text{AbsLevelPassX}[x_C][y_C] + \text{abs_remainder}[n])$	
}	
}	
}	

The number of context coded bins is restricted to be no larger than 2 bins per sample for each CG.

Table 9-15 – Assignment of ctxInc to syntax elements with context coded bins

Syntax element	binIdx					
	0	1	2	3	4	>= 5
last_sig_coeff_x_prefix	0..23 (clause 9.5.4.2.4)					
last_sig_coeff_y_prefix	0..23 (clause 9.5.4.2.4)					
last_sig_coeff_x_suffix	bypass	bypass	bypass	bypass	bypass	bypass
last_sig_coeff_y_suffix	bypass	bypass	bypass	bypass	bypass	bypass
coded_sub_block_flag[][]	$(\text{MaxCcbS} > 0) ? (0.7 \text{ (clause 9.5.4.2.6) }) : \text{bypass}$	na	na	na	na	na
sig_coeff_flag[][]	$(\text{MaxCcbS} > 0) ? (0.93 \text{ (clause 9.5.4.2.8) }) : \text{bypass}$	na	na	na	na	na
par_level_flag[]	$(\text{MaxCcbS} > 0) ? (0.33 \text{ (clause 9.5.4.2.9) }) : \text{bypass}$	na	na	na	na	na
abs_level_gtx_flag[][i]	0..70 (clause 9.5.4.2.9)	na	na	na	na	na
abs_remainder[]	bypass	bypass	bypass	bypass	bypass	bypass
dec_abs_level[]	bypass	bypass	bypass	bypass	bypass	bypass
coeff_sign_flag[] transform_skip_flag[x0][y0] == 0	bypass	na	na	na	na	na
coeff_sign_flag[] transform_skip_flag[x0][y0] == 1	0	na	na	na	na	na

3 Drawbacks of existing implementations

[00160] Several new intra prediction methods are supported in VVC. However, the current design of intra prediction methods has the following problems:

[00161] (1) The fixed order of ALWIP, MRL, ISP may be sub-optimal since it couldn't be adaptive to the local characteristics.

[00162] (2) The current design requires to pass several bins to reach the signaling of conventional intra prediction method. Apparently, the chance to select the conventional intra mode is relatively higher compared to all the other newly introduced methods in VVC, e.g., ALWIP, MRL, and ISP.

[00163] (3) MRL and ISP are only allowed for intra modes in MPM list, however, MRL and ISP related syntax are signaled even when the intra mode is not from MPM list. This may waste bits for the conventional intra prediction method.

4 Example methods for coding of multiple intra prediction methods

[00164] Embodiments of the presently disclosed technology overcome the drawbacks of existing implementations, thereby providing video coding with higher coding efficiencies. The methods for the coding of multiple intra prediction methods, based on the disclosed technology, may enhance both existing and future video coding standards, is elucidated in the following examples described for various implementations. The examples of the disclosed technology provided below explain general concepts, and are not meant to be interpreted as limiting. In an example, unless explicitly indicated to the contrary, the various features described in these examples may be combined.

[00165] In these examples, the conventional intra prediction method may represent the way that uses the adjacent line/column for intra prediction which may use interpolation filter along the prediction direction. And the additional intra coding methods may represent those which are newly introduced in VVC or may be introduced in the future and require additional signaling for the usage for this method. The additional method may be one or multiple of ALWIP, MRL, ISP, or QR-BDPCM/PCM etc.

1. The order of coding side information of the additional intra coding methods (e.g., ALWIP, MRL, ISP) may be changed from one video unit to another video unit.
 - a. In one example, the video unit is a sequence/view/picture/slice/tile/brick/CTU row/LCU/VPDU/CU/PU/TU etc. al.
 - b. In one example, indication of the coding order may be signaled in sequence/view/picture/slice/tile/brick/CTU row/LCU/VPDU/CU/PU/TU-level, etc. al.
 - c. In one example, how to select the order may depend on the block dimensions.

- d. In one example, the order may depend on the coded information from previously coded blocks.
 - i. In one example, a history table may be maintained and updated which may record the prediction methods of the most-recent several intra-coded blocks.
 - ii. In one example, the order may depend on the occurrence/frequency of each coding mode in the previously coded blocks.
 - iii. In one example, the order may depend on the coded methods of spatial (adjacent or non-adjacent) and/or temporal neighboring blocks.
 - e. In one example, the order may depend on whether one or multiple intra coding methods are applicable for the video unit. For example, for certain block dimension, one method may be disabled, therefore, there is no need to signal the side information of this method.
2. Indication of usage of conventional intra prediction method or additional intra coding methods (e.g., ALWIP, MRL, ISP) may be coded.
- a. In one example, the conventional intra prediction method may include the wide-angle intra-prediction.
 - i. Alternatively, the conventional intra prediction method may not include the wide-angle intra-prediction.
 - b. In one example, one-bit flag may be coded to indicate the conventional intra prediction method is used or not.
 - c. In one example, PCM mode flag may be coded before the indication (e.g., one-bit flag).
 - d. In one example, QR-BDPCM mode flag may be coded before the indication (e.g., one-bit flag).
 - e. In one example, additional intra coding methods may include QR-BDPCM.
 - i. Alternatively, additional intra coding methods may exclude QR-BDPCM
 - f. Alternatively, furthermore, syntax elements related to one or multiple of the additional intra coding methods may be coded after the indication of conventional intra prediction mode.

- g. In one example, one-bit flag may be coded to indicate whether the conventional intra prediction method or additional intra coding methods is used.
 - i. If the flag indicates the conventional intra prediction method is used, the signaling of the additional intra coding methods is skipped, and all the additional intra coding methods are inferred to be disabled.
 - ii. If the flag indicates the additional intra prediction methods are used, syntax elements related to the additional intra coding methods may be further coded in a given order.
 - 1) In one example, the order may be defined as ALWIP, MRL, ISP. Alternatively, any order of the three modes may be utilized.
 - 2) In one example, the order may be defined as QR-BDPCM, ALWIP, MRL, ISP. Alternatively, any order of the four modes may be utilized.
 - 3) In one example, the order may be defined as ALWIP, MRL, ISP, and other new coding methods.
 - 4) Alternatively, the order may be adaptively changed.
 - 5) Alternatively, furthermore, signaling of usage of the last intra coding method may be skipped.
 - a. For example, if the coding order is ALWIP, MRL and ISP. And the decoded syntax elements indicate ALWIP, MRL are both disabled, then signaling of the usage of ISP mode (e.g., **intra_subpartitions_mode_flag**) may be skipped.
 - b. In usage of the last intra coding method may be inferred if the signaling is skipped.
- h. The one-bit flag may be context coded with one or multiple contexts in the arithmetic coding.
 - i. In one example, multiple contexts may be allocated, one of them may be selected according to the information of neighboring blocks, such as whether the neighboring block is coded with conventional intra prediction method.

- ii. In one example, one or multiple contexts may be selected based on the bin index of a bin.
 - i. The one-bit flag may be conditionally signaled.
 - i. In one example, the flag may be not coded when all of the additional methods are disabled in one slice/tile group/tile/brick/picture/sequence.
 - ii. In one example, the flag may be not coded according to the block dimension.
 - j. Whether to follow the prior design or the proposed method may be signaled in sequence/picture/slice/tile group/tile/brick/CU/video data unit level.
 - i. In one example, whether to enable the proposed method may signaled in DPS/SPS/VPS/PPS/APS/picture header/slice header/tile group header etc. al.
3. The MPM list construction process may be unified for all the additional intra coding methods (e.g, MRL, ISP and ALWIP), i.e., following the same logic to derive multiple intra prediction mode candidates.
- a. Alternatively, the MPM list construction process may be unified for all the additional intra coding methods (e.g, MRL, ISP, ALWIP, and QR-BDPCM).
 - b. Alternatively, an additional stage may be invoked to derive the modified intra prediction mode candidates for one additional intra coding method, such as reordering or discarding a mode, or replacing one mode in the MPM list by another one.
 - c. Alternatively, furthermore, for each coding method, a given set of allowed intra prediction modes may be pre-defined/signaled.
 - d. Alternatively, furthermore, the modes in the MPM list may be further mapped to one of allowed intra prediction modes in the given set for one additional intra coding method.
 - e. Alternatively, furthermore, different intra coding method may select partial or full of modes in the MPM list.
4. One or multiple additional intra coding methods (e.g., MRL or/and ISP or/and QR-BDPCM) may be allowed only for certain intra-prediction modes.

- a. In one example, one or multiple of additional intra coding methods (e.g., MRL or/and ISP or/and QR-BDPCM) may be allowed for the K ($K > 0$) intra-prediction modes in the MPM list. For example, $K = 2$ or 3 . In one example, the K intra-prediction modes may be the first K intra-prediction mode in the MPM list.
 - b. In one example, one or multiple additional intra coding methods (e.g., MRL or/and ISP or/and QR-BDPCM) may be allowed for pre-defined sets of intra-prediction modes, e.g., Planar, DC, vertical, horizontal.
 - c. In one example, the first K intra modes in the MPM list may be checked in order, and when an intra mode in the MPM list is also part of a predefined/signaled intra-prediction mode set, the one or multiple of additional intra-prediction coding methods (e.g., MRL or/and ISP or/and QR-BDPCM) may be allowed for it.
 - d. In one example, intra-prediction modes in the MPM list may be checked in order until K valid (when an intra-prediction mode is included in a predefined intra mode set, it is considered as valid) intra-prediction modes are found or all modes in the MPM list are checked.
5. Intra-prediction modes information may be signaled before the usage of one or multiple additional intra coding methods (e.g., MRL or/and ISP).
- a. Alternatively, furthermore, the usage of one or multiple additional intra coding methods (e.g., MRL or/and ISP) may be conditionally signaled according to the intra-prediction mode information.
 - b. In one example, if the decoded intra-prediction mode is not allowed for one additional intra coding method, the usage of such method is not signaled any more.
 - c. In one example, all of additional intra coding methods (e.g., MRL, ISP, ALWIP) may be coded after the signaling of intra-prediction modes (e.g., **intra_luma_mpm_flag** or **intra_luma_not_planar_flag** or **intra_luma_mpm_idx**, or **intra_luma_mpm_remainder**).
 - d. Alternatively, some of additional intra coding methods (e.g., MRL, ISP) may be coded after the signaling of intra prediction modes (e.g.,

intra_luma_mpm_flag or **intra_luma_not_planar_flag** or **intra_luma_mpm_idx**, or **intra_luma_mpm_remainder**) and the remaining (e.g., QR-BDPCM and/or ALWIP) may be coded before the signaling of intra prediction modes.

- i. Alternatively, indications of usage of QR-BDPCM, MRL and ALWIP may be coded before intra-prediction modes, while ISP may be coded after.
 - e. In one example, the signaling of usage of one or multiple additional intra coding methods (e.g., MRL or/and ISP) may depend on whether the intra-prediction mode corresponds to the wide-angle intr-prediction.
6. MRL or/and ISP or/and other methods related syntax elements may be signaled after certain intra coding information, and whether to signal them or not may depend on the intra coding information.
- a. In one example, the intra coding information may include whether the intra-prediction mode is from the MPM list (e.g., **intra_luma_mpm_flag**).
 - i. In one example, side information of ISP or/and MRL or/and other methods (e.g., **intra_luma_ref_idx**, **intra_subpartitions_mode_flag**, **intra_subpartitions_split_flag**) may be conditional signaled after/right after the MPM flag (e.g., **intra_luma_mpm_flag**).
 - ii. Alternatively, furthermore, the intra coding information may include whether the intra-prediction mode is planar or not (e.g., **intra_luma_not_planar_flag**) or whether it is the first MPM candidate or not.
 - 1) Alternatively, furthermore, side information of ISP or/and MRL or/and other methods (e.g., **intra_luma_ref_idx**, **intra_subpartitions_mode_flag**, **intra_subpartitions_split_flag**) may be conditionally signaled after/right after the planar mode flag/first MPM flag (e.g., **intra_luma_not_planar_flag**).
 - iii. Alternatively, furthermore, the intra coding information may include the remaining MPM index (e.g., **intra_luma_mpm_idx**).

- 1) Alternatively, furthermore, side information of ISP or/and MRL or/and other methods (e.g., **intra_luma_ref_idx**, **intra_subpartitions_mode_flag**, **intra_subpartitions_split_flag**) may be conditionally signaled after/right after remaining MPM index (e.g., **intra_luma_mpm_idx**).
- iv. Alternatively, furthermore, the intra coding information may include the intra-prediction mode (e.g., **intra_luma_not_planar_flag**, **intra_luma_mpm_idx**, **intra_luma_mpm_remainder**).
 - 1) Alternatively, furthermore, side information of ISP or/and MRL or/and other methods (e.g., **intra_luma_ref_idx**, **intra_subpartitions_mode_flag**, **intra_subpartitions_split_flag**) may be conditionally signaled after/right after the intra prediction mode.
- b. In one example, when intra-prediction mode of the block is not from the MPM list (e.g., **intra_luma_mpm_flag** is equal to 0), MRL or/and ISP or/and other methods related syntax elements may not be signaled.
 - i. Alternatively, furthermore, the signaling of **intra_luma_mpm_flag** is independent from usages of ISP and MRL.
 - 1) In one example, **intra_luma_mpm_flag** is signaled without the conditional check of **intra_subpartitions_mode_flag** and **intra_luma_ref_idx** are both equal to 0.
 - ii. When one method related syntax is not signaled, such method is not applied to this block.
 - 1) In one example, ISP may not be applied for the block (e.g., **intra_subpartitions_mode_flag** is inferred to be 0).
 - 2) Alternatively, furthermore, MRL may not be allowed for the block (e.g., **intra_luma_ref_idx** is inferred to be 0).
- c. Alternatively, furthermore, signaling of MRL or/and ISP related syntax elements may be further dependent on the block dimensions.

- d. Alternatively, furthermore, MRL related syntax elements may not be signaled for planar mode.
 - i. In one example, when **intra_luma_not_planar_flag** is equal to 0, signaling of the usage of MRL (e.g., **intra_luma_ref_idx**) is skipped.
 - ii. Alternatively, furthermore, MRL may not be allowed for the block (e.g., **intra_luma_ref_idx** is inferred to be 0).
- 7. A candidate intra coding method list (denoted by IPMList) may be constructed, and an index to the list may be coded.
 - a. *IPMIdx* associated with spatial neighboring blocks (adjacent or non-adjacent) may be inserted to the IPMList in order.
 - b. In one example, conventional intra coding method may be the first candidate in the IPMList.
 - c. In one example, the spatial neighboring blocks may be defined to be those which are used in the motion candidate list construction process of inter-AMVP code/merge mode/affine mode/IBC mode or the MPM list construction process in the normal intra prediction method.
 - d. In one example, the checking order of spatial neighboring blocks may be defined to be the same or different as that used in the motion candidate list construction process of inter-AMVP code/merge mode/affine mode/IBC mode or the MPM list construction process in the normal intra prediction method.
 - e. Alternatively, furthermore, the inserted *IPMIdx* to the list may be further refined.
 - f. *IPMIdx* may be binarized with truncated unary code /k-th order EG code /fixed length code.
- 8. Indications of multiple intra coding methods may be jointly coded, such as by one syntax element.
 - a. In one example, the one syntax element may be used to indicate the selected intra coding method (e.g., conventional intra, ALWIP, MRL, ISP).
 - b. In one example, an intra coding method may be represented by an index (denoted by *IPMIdx*) and the corresponding index of one selected coding method for a block may be coded.

- c. In one example, the syntax element may be binarized with truncated unary/k-th order EG/fixed length.
 - d. In one example, bins of the bin string for the syntax element may context-coded or bypass coded.
 - e. In one example, the semantics of the syntax element (i.e., the mapping of the decoded value and the intra prediction method) may be changed from one video unit to another.
9. Whether to enable or disable the above methods may be signaled in DPS/SPS/PPS/APS/VPS/sequence header/picture header/slice header/tile group header/tile/group of CTUs, etc.
- a. Alternatively, which method to be used may be signaled in DPS/SPS/PPS/APS/VPS/sequence header/picture header/slice header/tile group header/ tile/group of CTUs, etc.
 - b. Alternatively, whether to enable or disable the above methods and/or which method to be applied may be dependent on block dimension, Virtual Pipelining Data Units (VPDU), picture type, low delay check flag.
 - c. Alternatively, whether to enable or disable the above methods and/or which method to be applied may be dependent on the color component, color format, etc.

[00166] The examples described above may be incorporated in the context of the methods described below, e.g., methods 2100, 2110, 2120, 2130 and 2140, which may be implemented at a video decoder or a video encoder.

[00167] FIG. 21A shows a flowchart of an exemplary method for video processing. The method 2100 includes, at step 2102, configuring an order of coding side information of one or more intra coding methods for a current video block different from a previous order of coding side information for a previous video block.

[00168] The method 2100 includes, at step 2104, performing, based on the configuring, a conversion between the current video block and a bitstream representation of the current video block.

[00169] In some embodiments, the one or more intra coding methods comprises at least one of

an affine linear weighted intra prediction (ALWIP) method, a multiple reference line (MRL) intra prediction method or an intra subblock partitioning (ISP) method.

[00170] In some embodiments, an indication of the order is signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a slice header, a tile header, a coding tree unit (CTU) row, a largest coding unit (LCU) or a virtual pipelining data unit (VPDU). In an example, the order is based on a height and/or a width of the current video block. In another example, the order is based on coded information of the previous video block. In yet another example, the order is based on an applicability of the one or more intra coding methods to the current video block.

[00171] FIG. 21B shows a flowchart of another exemplary method for video processing. The method 2110 includes, at step 2112, making a decision that a current video block is coded using a coding method selected from a group comprising an affine linear weighted intra prediction (ALWIP) method, a multiple reference line (MRL) intra prediction method, a quantized residual block differential pulse-code modulation (QR-BDPCM) coding method and an intra subblock partitioning (ISP) method.

[00172] The method 2110 includes, at step 2114, constructing, based on the decision, a most probable mode (MPM) list for the coding method based on a construction process that is common for each coding method in the group.

[00173] The method 2110 includes, at step 2116, performing, based on the MPM list, a conversion between the current video block and a bitstream representation of the current video block.

[00174] In some embodiments, a set of allowed intra prediction modes for each coding method in the group is signaled in the bitstream representation.

[00175] In some embodiments, one or more coding methods in the group is allowed for K intra prediction modes in the MPM list, and K is a positive integer (e.g., $K = 2, 3$). In an example, the K intra prediction modes in the MPM list are the first K intra prediction modes.

[00176] FIG. 21C shows a flowchart of yet another exemplary method for video processing. The method 2120 includes, at step 2122, configuring, for an application of one or more intra coding methods to a current video block, a bitstream representation of the current video block, wherein at least one of (i) an indication of the application, (ii) syntax elements related to the one or more intra coding methods, (iii) information related to intra-prediction modes or (iv) intra

coding information is signaled in the bitstream representation.

[00177] The method 2120 includes, at step 2124, performing, based on the configuring, a conversion between the current video block and the bitstream representation of the current video block. In some embodiments, the indication of the application is signaled before the information related to intra-prediction modes.

[00178] In some embodiments, the syntax elements related to the one or more intra coding methods is signaled after the intra coding information. In an example, the intra coding information comprises a determination as to whether at least one of the intra-prediction modes is from a most probable mode (MPM) list. In another example, an inclusion of the syntax elements related to the one or more intra coding methods in the bitstream representation is based on at least one dimension of the current video block.

[00179] FIG. 21D shows a flowchart of yet another exemplary method for video processing. The method 2130 includes, at step 2132, configuring, for an application of one or more intra coding methods to a current video block, a bitstream representation of the current video block, wherein a single syntax element that jointly codes the application of the one or more intra coding methods is signaled in the bitstream representation.

[00180] The method 2130 includes, at step 2134, performing, based on the configuring, a conversion between the current video block and the bitstream representation of the current video block.

[00181] In some embodiments, the single syntax element comprises a selection of at least one of the one or more intra coding methods.

[00182] In some embodiments, the single syntax element is binarized with a truncated unary code, an exponential-Golomb code of order K or a fixed length code.

[00183] In some embodiments, a plurality of bins of a bin string for the single syntax element is context coded or bypass coded.

[00184] In some embodiments, semantics of the single syntax element for the current video block are different from semantics of the single syntax element for a previous video block.

[00185] FIG. 21E shows a flowchart of yet another exemplary method for video processing. The method 2140 includes, at step 2142, configuring, for an application of a conventional intra prediction method or one or more intra coding methods to a current video block, a bitstream representation of the current video block, wherein an indication of the application is coded in the

bitstream representation.

[00186] The method 2140 includes, at step 2144, performing, based on the configuring, a conversion between the current video block and the bitstream representation of the current video block. In some embodiments, the one or more intra coding methods comprises at least one of an affine linear weighted intra prediction (ALWIP) method, a multiple reference line (MRL) intra prediction method or an intra subblock partitioning (ISP) method.

[00187] In some embodiments, the conventional intra prediction method comprises a wide-angle intra prediction mode. In other embodiments, the conventional intra prediction method excludes a wide-angle intra prediction mode.

[00188] In some embodiments, the one or more intra coding methods comprises a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode. In other embodiments, the one or more intra coding methods excludes a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.

[00189] In some embodiments, the application of the conventional intra prediction method or the one or more intra coding methods is coded using a one-bit flag. In an example, a pulse code modulation (PCM) mode flag is coded before the one-bit flag. In another example, a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode flag is coded before the one-bit flag. In yet another example, at least one syntax element related to the one or more intra coding methods is coded after the one-bit flag. In yet another example, the one-bit flag is context coded with one or more arithmetic coding contexts.

[00190] In some embodiments, the bitstream representation further comprises syntax elements related to the one or more intra coding methods that are coded in an order.

[00191] In some embodiments, the methods 2100, 2110, 2120, 2130 and 2140 further include the step of making a decision, for the current video block, regarding a selective application of at least one of the one or more intra coding methods. In an example, the decision is signaled in a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs). In another example, the decision is based on at least one dimension of the current video block, a virtual data pipelining unit (VDPU) picture type or a low delay check flag. In yet another example, the decision is based on a color component or a color format of the current video block.

5 Example implementations of the disclosed technology

[00192] Improvement of the signaling of additional intra coding methods (taking MRL and ISP related syntax elements for example) are presented in this section.

[00193] In the following exemplary embodiments, the changes are highlighted in grey. Deleted texts are marked with double brackets (e.g., [[a]] denotes the deletion of the character “a”).

5.1 Embodiment #1

[00194] Signaling of additional intra coding methods (e.g., both MRL and ISP) are after the signaling of MPM flag.

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
if(slice_type != I sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag[x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && slice_type != I)	
pred_mode_flag	ae(v)
if(((slice_type == I && cu_skip_flag[x0][y0] == 0) (slice_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] == MODE_INTRA) {	
if(sps_pcm_enabled_flag && cbWidth >= MinIpcmCbSizeY && cbWidth <= MaxIpcmCbSizeY && cbHeight >= MinIpcmCbSizeY && cbHeight <= MaxIpcmCbSizeY)	
pcm_flag[x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned())	
pcm_alignment_zero_bit	f(1)
pcm_sample(cbWidth, cbHeight, treeType)	
} else {	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_LUMA) {	
[[if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))	
intra_subpartitions_mode_flag[x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_subpartitions_split_flag[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] == 0 && intra_subpartitions_mode_flag[x0][y0] == 0) [[	
intra_luma_mpm_flag[x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0]) {	
if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx[x0][y0]	ae(v)

if(intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))	
intra_subpartitions_mode_flag [x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_subpartitions_split_flag [x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] == 0)	
intra_luma_not_planar_flag [x0][y0]	ae(v)
if(intra_luma_not_planar_flag[x0][y0])	
intra_luma_mpm_idx [x0][y0]	ae(v)
} else	
intra_luma_mpm_remainder [x0][y0]	ae(v)
}	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_CHROMA)	
intra_chroma_pred_mode [x0][y0]	ae(v)
}	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
...	
}	
...	
}	

5.2 Embodiment #2

[00195] Signaling of one additional intra coding method (e.g., ISP) are after the signaling of MPM flag.

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
if(slice_type != I sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag [x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && slice_type != I)	
pred_mode_flag	ae(v)
if(((slice_type == I && cu_skip_flag[x0][y0] == 0) (slice_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] == MODE_INTRA) {	
if(sps_pcm_enabled_flag && cbWidth >= MinIpcmCbSizeY && cbWidth <= MaxIpcmCbSizeY && cbHeight >= MinIpcmCbSizeY && cbHeight <= MaxIpcmCbSizeY)	
pcm_flag [x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned())	
pcm_alignment_zero_bit	f(1)
pcm_sample(cbWidth, cbHeight, treeType)	
} else {	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_LUMA) {	
if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx [x0][y0]	ae(v)

<pre> if(intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY)) </pre>	
<pre> intra_subpartitions_mode_flag[x0][y0] </pre>	ae(v)
<pre> if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY) </pre>	
<pre> intra_subpartitions_split_flag[x0][y0] </pre>	ae(v)
<pre> if(intra_luma_ref_idx[x0][y0] == 0 && intra_subpartitions_mode_flag[x0][y0] == 0) </pre>	
<pre> if(intra_luma_ref_idx[x0][y0] == 0) </pre>	
<pre> intra_luma_mpm_flag[x0][y0] </pre>	ae(v)
<pre> if(intra_luma_mpm_flag[x0][y0]) { </pre>	
<pre> if(intra_luma_ref_idx[x0][y0] == 0 && (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY)) </pre>	
<pre> intra_subpartitions_mode_flag[x0][y0] </pre>	ae(v)
<pre> if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY) </pre>	
<pre> intra_subpartitions_split_flag[x0][y0] </pre>	ae(v)
<pre> if(intra_luma_ref_idx[x0][y0] == 0) </pre>	
<pre> intra_luma_not_planar_flag[x0][y0] </pre>	ae(v)
<pre> if(intra_luma_not_planar_flag[x0][y0]) </pre>	
<pre> intra_luma_mpm_idx[x0][y0] </pre>	ae(v)
<pre> } else </pre>	
<pre> intra_luma_mpm_remainder[x0][y0] </pre>	ae(v)
<pre> } </pre>	
<pre> if(treeType == SINGLE_TREE treeType == DUAL_TREE_CHROMA) </pre>	
<pre> intra_chroma_pred_mode[x0][y0] </pre>	ae(v)
<pre> } </pre>	
<pre> } else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */ </pre>	
<pre> ... </pre>	
<pre> } </pre>	
<pre> ... </pre>	
<pre> } </pre>	

5.3 Embodiment #3

[00196] Signaling of one additional intra coding method (e.g., MRL) are after the signaling of MPM flag.

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
if(slice_type != I sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag [x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && slice_type != I)	
pred_mode_flag	ae(v)
if(((slice_type == I && cu_skip_flag[x0][y0] == 0)	
(slice_type != I && CuPredMode[x0][y0] != MODE_INTRA)) &&	
sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] == MODE_INTRA) {	

if(sps_pcm_enabled_flag && cbWidth >= MinIpcmCbSizeY && cbWidth <= MaxIpcmCbSizeY && cbHeight >= MinIpcmCbSizeY && cbHeight <= MaxIpcmCbSizeY)	
pcm_flag [x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned())	
pcm_alignment_zero_bit	f(1)
pcm_sample(cbWidth, cbHeight, treeType)	
} else {	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_LUMA) {	
[[if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx [x0][y0]]	[[ae(v)]]
if([[intra_luma_ref_idx[x0][y0] == 0 &&] (cbWidth <= MaxTbSizeY cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))	
intra_subpartitions_mode_flag [x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] == 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_subpartitions_split_flag [x0][y0]	ae(v)
if([[intra_luma_ref_idx[x0][y0] == 0 &&] intra_subpartitions_mode_flag[x0][y0] == 0)	
intra_luma_mpm_flag [x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0]) {	
if((y0 % CtbSizeY) > 0 && intra_subpartitions_mode_flag[x0][y0] == 0)	
intra_luma_ref_idx [x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] == 0)	
intra_luma_not_planar_flag [x0][y0]	ae(v)
if(intra_luma_not_planar_flag[x0][y0])	
intra_luma_mpm_idx [x0][y0]	ae(v)
} else	
intra_luma_mpm_remainder [x0][y0]	ae(v)
}	
if(treeType == SINGLE_TREE treeType == DUAL_TREE_CHROMA)	
intra_chroma_pred_mode [x0][y0]	ae(v)
}	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
...	
}	
...	
}	

[00197] FIG. 22 is a block diagram of a video processing apparatus 2200. The apparatus 2200 may be used to implement one or more of the methods described herein. The apparatus 2200 may be embodied in a smartphone, tablet, computer, Internet of Things (IoT) receiver, and so on. The apparatus 2200 may include one or more processors 2202, one or more memories 2204 and video processing hardware 2206. The processor(s) 2202 may be configured to implement one or more methods (including, but not limited to, methods 2100, 2110, 2120, 2130 and 2140) described in the present document. The memory (memories) 2204 may be used for storing data

and code used for implementing the methods and techniques described herein. The video processing hardware 2206 may be used to implement, in hardware circuitry, some techniques described in the present document.

[00198] In some embodiments, the video coding methods may be implemented using an apparatus that is implemented on a hardware platform as described with respect to FIG. 22.

[00199] FIG. 23 is a flowchart for an example method 2300 of video processing. The method 2300 includes, at 2302, determining, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, an order of coding side information of additional intra coding methods associated with the video processing unit, wherein the order is changed from one video processing unit to another video processing unit; and at 2304, performing the conversion based on the determined order.

[00200] In some examples, the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method.

[00201] In some examples, the additional intra coding methods include at least one of an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode or an intra subblock partitioning (ISP) mode.

[00202] In some examples, the video processing unit is one of a sequence, view, picture, slice, tile, brick, coding tree unit (CTU) row, largest coding unit (LCU), Virtual Pipelining Data Units (VPDU), coding unit (CU), prediction unit (PU) and transform unit (TU).

[00203] In some examples, an indication of the order is signaled in at least one of a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a slice header, a tile header, a coding tree unit (CTU) row, a largest coding unit (LCU) or a virtual pipelining data unit (VPDU).

[00204] In some examples, the order depends on dimensions of the video processing unit.

[00205] In some examples, the order is based on coded information of one or multiple previous coded video processing units.

[00206] In some examples, a history table is maintained and updated which records prediction methods of the multiple previous coded video processing units which are the most-recent intra-coded video processing units.

[00207] In some examples, the order depends on occurrence and/or frequency of each coding mode in the previously coded video processing units.

[00208] In some examples, the order depends on coded methods of spatial adjacent or non-adjacent and/or temporal neighboring coded video processing units of the video processing unit.

[00209] In some examples, the order depends on whether one or multiple intra coding methods are applicable for the video processing unit.

[00210] In some examples, for certain dimension of the video processing unit, certain method is disabled, and the side information of the certain method is not signaled.

[00211] FIG. 24 is a flowchart for an example method 2400 of video processing. The method 2400 includes, at 2402, coding, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, an indication of usage of additional intra coding methods or conventional intra prediction method for the video processing unit in the bitstream, wherein the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method, and the conventional intra prediction method includes one or more intra coding methods that use adjacent line or column for intra prediction which uses interpolation filter along prediction direction; at 2404, performing the conversion based on the indication.

[00212] In some examples, the additional intra coding methods includes at least one of an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode or an intra subblock partitioning (ISP) mode.

[00213] In some examples, the conventional intra prediction method includes a wide-angle intra-prediction method.

[00214] In some examples, the conventional intra prediction method does not include a wide-angle intra-prediction method.

[00215] In some examples, a one-bit flag is coded to indicate the conventional intra prediction method is used or not.

[00216] In some examples, a pulse code modulation (PCM) mode flag is coded before the indication, where the PCM mode flag is a one-bit flag.

[00217] In some examples, a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode flag is coded before the indication, where the QR-BDPCM mode flag is a one-bit flag.

[00218] In some examples, the additional intra coding methods include a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.

[00219] In some examples, the additional intra coding methods exclude a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.

[00220] In some examples, syntax elements related to one or multiple of the additional intra coding methods are coded after the indication of the conventional intra prediction mode.

[00221] In some examples, a one-bit flag is coded to indicate whether the conventional intra prediction method or the additional intra coding methods is used.

[00222] In some examples, if the flag indicates the conventional intra prediction method is used, the signaling of the additional intra coding methods is skipped, and all the additional intra coding methods are inferred to be disabled.

[00223] In some examples, if the flag indicates the additional intra prediction methods are used, syntax elements related to the additional intra coding methods are further coded in a given order.

[00224] In some examples, the order is defined as ALWIP mode, MRL mode and ISP mode.

[00225] In some examples, any order of ALWIP mode, MRL mode and ISP mode is utilized as the given order.

[00226] In some examples, the order is defined as QR-BDPCM mode, ALWIP mode, MRL mode and ISP mode.

[00227] In some examples, any order of QR-BDPCM mode, ALWIP mode, MRL mode and ISP mode is utilized as the given order.

[00228] In some examples, the order is defined as ALWIP mode, MRL mode, ISP mode and other new coding methods.

[00229] In some examples, the order is adaptively changed.

[00230] In some examples, signaling of usage of the last intra coding method is skipped.

[00231] In some examples, if the order is ALWIP mode, MRL mode and ISP mode, decoded syntax elements indicate ALWIP mode and MRL mode are both disabled, signaling of the usage of ISP mode indicated by `intra_subpartitions_mode_flag` is skipped.

[00232] In some examples, if the signaling is skipped, the usage of the last intra coding method is inferred.

[00233] In some examples, the one-bit flag is context coded with one or multiple contexts in arithmetic coding.

[00234] In some examples, when multiple contexts are allocated, one of the multiple contexts

is selected according to information of neighboring video processing units including information of whether the neighboring video processing unit is coded with the conventional intra prediction method.

[00235] In some examples, the one or multiple contexts is selected based on bin index of a bin.

[00236] In some examples, the one-bit flag is conditionally signaled.

[00237] In some examples, when all of the additional intra coding methods are disabled in at least one of one slice, tile group, tile, brick, picture and sequence, the one-bit flag is not coded.

[00238] In some examples, the one-bit flag is not coded according to dimension of the video processing unit.

[00239] In some examples, whether to follow prior design or the coding of the indication is signaled in at least one of sequence, picture, slice, tile group, tile, brick, CU and video data unit level.

[00240] In some examples, whether to enable the coding of the indication is signaled in at least one of a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs).

[00241] FIG. 25 is a flowchart for an example method 2500 of video processing. The method 2500 includes, at 2502, constructing, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, a most probable mode (MPM) list by unifying a MPM list construction process for all additional intra coding methods associated with the video processing unit; and at 1004, performing the conversion based on the MPM list.

[00242] In some examples, the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method.

[00243] In some examples, the additional intra coding methods include an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode and an intra subblock partitioning (ISP) mode.

[00244] In some examples, the additional intra coding methods include an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode, an intra subblock partitioning (ISP) mode, and a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.

[00245] In some examples, an additional stage is invoked to derive modified intra prediction mode candidates for one additional intra coding method.

[00246] In some examples, the additional stage is invoked to reorder the MPM list, or discard a mode, or replacing one mode in the MPM list by another one.

[00247] In some examples, for each coding methods, a given set of allowed intra prediction modes is pre-defined and/or signaled.

[00248] In some examples, modes in the MPM list are further mapped to one of allowed intra prediction modes in the given set for one additional intra coding method.

[00249] In some examples, different intra coding method selects partial or full of modes in the MPM list.

[00250] In some examples, one or multiple of the additional intra coding methods are allowed only for certain intra-prediction modes.

[00251] In some examples, the one or more of the additional intra coding methods include MRL mode, or/and ISP mode or/and QR-BDPCM mode.

[00252] In some examples, one or multiple of the additional intra coding methods are allowed for K intra-prediction modes in the MPM list, K being an integer.

[00253] In some examples, K is 2 or 3.

[00254] In some examples, the K intra-prediction modes are the first K intra-prediction mode in the MPM list.

[00255] In some examples, one or multiple of the additional intra coding methods are allowed for pre-defined sets of intra-prediction modes including Planar, DC, vertical, horizontal mode.

[00256] In some examples, a first K intra modes in the MPM list are checked in order, and when an intra mode in the MPM list is also part of a predefined or signaled intra-prediction mode set, the one or multiple of the additional intra-prediction coding methods are allowed for the intra mode, K being an integer.

[00257] In some examples, intra-prediction modes in the MPM list are checked in order until K valid intra-prediction modes are found or all modes in the MPM list are checked, wherein when the intra-prediction mode is included in a predefined intra mode set, the intra-prediction mode is considered as valid.

[00258] FIG. 26 is a flowchart for an example method 2600 of video processing. The method 2600 includes, at 2602, signaling, for a conversion between a video processing unit of the video

and a bitstream representation of the video processing unit, intra-prediction modes information before usage of one or multiple additional intra coding methods associated with the video processing unit; and at 2604, performing the conversion based on the intra-prediction modes information.

[00259] In some examples, the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method.

[00260] In some examples, the one or multiple additional intra coding methods include a multiple reference line (MRL) intra prediction mode and/or an intra subblock partitioning (ISP) mode.

[00261] In some examples, the usage of one or multiple additional intra coding methods is conditionally signaled according to the intra-prediction mode information.

[00262] In some examples, if decoded intra-prediction mode is not allowed for one additional intra coding method, the usage of the one additional intra coding method is not signaled any more.

[00263] In some examples, all of the additional intra coding methods including MRL mode, ISP mode and ALWIP mode are coded after the signaling of intra-prediction modes including `intra_luma_mpm_flag`, `intra_luma_not_planar_flag`, `intra_luma_mpm_idx` or `intra_luma_mpm_remainder`.

[00264] In some examples, partial of the additional intra coding methods including MRL mode and ISP mode are coded after the signaling of intra-prediction modes including `intra_luma_mpm_flag`, `intra_luma_not_planar_flag`, `intra_luma_mpm_idx` or `intra_luma_mpm_remainder`, and remaining of the additional intra coding methods including QR-BDPCM mode and/or ALWIP mode is coded before the signaling of intra prediction modes.

[00265] In some examples, indications of usage of the additional intra coding methods including QR-BDPCM mode, MRL mode and ALWIP mode is coded before the signaling of intra-prediction modes, while ISP mode is coded after the signaling of intra prediction modes.

[00266] In some examples, the signaling of usage of the one or multiple additional intra coding methods depends on whether the intra-prediction mode corresponds to wide-angle intra-prediction.

[00267] FIG. 27 is a flowchart for an example method 2700 of video processing. The method 2700 includes, at 2702, signaling, for a conversion between a video processing unit of the video

and a bitstream representation of the video processing unit, one or multiple additional intra coding methods related syntax elements after certain intra coding information; and at 2704, performing the conversion based on the one or multiple additional intra coding methods related syntax elements.

[00268] In some examples, the one or multiple additional intra coding methods include a multiple reference line (MRL) intra prediction mode and/or an intra subblock partitioning (ISP) mode and/or other methods.

[00269] In some examples, whether signaling the one or multiple additional intra coding methods related syntax elements depends on the intra coding information.

[00270] In some examples, the intra coding information includes a MPM flag indicating whether an intra-prediction mode for the video processing unit is from a most probable mode (MPM) list, wherein the MPM flag is denoted by `intra_luma_mpm_flag`.

[00271] In some examples, side information of the one or multiple additional intra coding methods is conditional signaled after or right after the MPM flag, wherein the side information includes at least one of `intra_luma_ref_idx`, `intra_subpartitions_mode_flag`, `intra_subpartitions_split_flag`.

[00272] In some examples, the intra coding information includes a planar mode flag indicating whether an intra-prediction mode for the video processing unit is planar or not, or a first MPM flag indicating whether the intra-prediction mode is a first MPM candidate in the MPM list or not, wherein the planar mode flag is denoted by `intra_luma_not_planar_flag`.

[00273] In some examples, side information of the one or multiple additional intra coding methods is conditional signaled after or right after the planar mode flag or the first MPM flag.

[00274] In some examples, the intra coding information includes the remaining MPM index which is denoted by `intra_luma_mpm_idx`.

[00275] In some examples, side information of the one or multiple additional intra coding methods is conditional signaled after or right after the remaining MPM index.

[00276] In some examples, the intra coding information includes intra-prediction mode which is indicated by at least one of `intra_luma_not_planar_flag`, `intra_luma_mpm_idx` and `intra_luma_mpm_remainder`.

[00277] In some examples, side information of the one or multiple additional intra coding methods is conditional signaled after or right after the intra-prediction mode.

[00278] In some examples, when intra-prediction mode of the block is not from the MPM list with the MPM flag `intra_luma_mpm_flag` is equal to 0, the one or multiple additional intra coding methods related syntax elements are not signaled.

[00279] In some examples, the signaling of the MPM flag `intra_luma_mpm_flag` is independent from usages of ISP mode and MRL mode.

[00280] In some examples, the MPM flag `intra_luma_mpm_flag` is signaled without conditional check of `intra_subpartitions_mode_flag` and `intra_luma_ref_idx` are both equal to 0.

[00281] In some examples, when one method related syntax is not signaled, the method is not applied to the video processing unit.

[00282] In some examples, when `intra_subpartitions_mode_flag` is inferred to be 0, ISP mode is not applied for the video processing unit.

[00283] In some examples, when `intra_luma_ref_idx` is inferred to be 0, MRL mode is not applied for the video processing unit.

[00284] In some examples, signaling of MRL or/and ISP related syntax elements is further dependent on dimensions of the video processing unit.

[00285] In some examples, MRL related syntax elements is not signaled for planar mode.

[00286] In some examples, when `intra_luma_not_planar_flag` is equal to 0, signaling of the usage of MRL is skipped.

[00287] In some examples, when `intra_luma_ref_idx` is inferred to be 0, MRL mode is not applied for the video processing unit.

[00288] FIG. 28 is a flowchart for an example method 2800 of video processing. The method 2800 includes, at 2802, constructing, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, candidate intra coding method list (IPMList) for the video processing unit; and at 2804, performing the conversion based on the IPMList.

[00289] In some examples, an index `IPMIdx` to the IPMList is coded.

[00290] In some examples, `IPMIdx` associated with spatial neighboring video processing units is inserted to the IPMList in order, where the spatial neighboring video processing units include spatial neighboring adjacent or non-adjacent video processing units.

[00291] In some examples, conventional intra coding method is a first candidate in the IPMList, wherein the conventional intra prediction method includes one or more intra coding

methods that use adjacent line or column for intra prediction which uses interpolation filter along prediction direction.

[00292] In some examples, the spatial neighboring video processing units are defined to be those which are used in motion candidate list construction process of inter-AMVP code, merge mode, affine mode, IBC mode or the MPM list construction process in the normal intra prediction method.

[00293] In some examples, checking order of the spatial neighboring blocks are defined to be the same or different as that used in the motion candidate list construction process of inter-AMVP code, merge mode, affine mode, IBC mode or the MPM list construction process in the normal intra prediction method.

[00294] In some examples, the inserted IPMIdx to the list is further refined.

[00295] In some examples, IPMIdx is binarized with truncated unary code, k-th order EG code or fixed length code.

[00296] FIG. 29 is a flowchart for an example method 2900 of video processing. The method 2900 includes, at 2902, jointly coding, for a conversion between a video processing unit of the video and a bitstream representation of the video processing unit, indications of multiple intra coding methods associated with the video processing unit by using one syntax element; and at 2904, performing the conversion based on the coded indications.

[00297] In some examples, the one syntax element is used to indicate a selected intra coding method.

[00298] In some examples, the selected intra coding method includes at least one of conventional intra coding method and additional intra coding methods including an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode and an intra subblock partitioning (ISP) mode.

[00299] In some examples, an intra coding method is represented by an index IPMIdx and the corresponding index of one selected coding method for the video processing unit is coded.

[00300] In some examples, the syntax element is binarized with truncated unary, k-th order EG or fixed length.

[00301] In some examples, bins of a bin string for the syntax element is context-coded or bypass coded.

[00302] In some examples, semantics of the syntax element which indicates a mapping of

decoded value and the intra prediction method is changed from one video processing unit to another.

[00303] In some examples, whether to enable or disable the determining process, the coding process, the constructing process, the signaling process or the jointly coding process is signaled in at least one of a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs).

[00304] In some examples, which process of the determining process, the coding process, the constructing process, the signaling process or the jointly coding process is to be used is signaled in at least one of a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs).

[00305] In some examples, whether to enable or disable the processes and/or which process is to be used depends on at least one of dimension of the video processing unit, Virtual Pipelining Data Units (VPDU), picture type and low delay check flag.

[00306] In some examples, whether to enable or disable the processes and/or which process is to be used depends on at least one of color component and color format.

[00307] From the foregoing, it will be appreciated that specific embodiments of the presently disclosed technology have been described herein for purposes of illustration, but that various modifications may be made without deviating from the scope of the invention. Accordingly, the presently disclosed technology is not limited except as by the appended claims.

[00308] Implementations of the subject matter and the functional operations described in this patent document can be implemented in various systems, digital electronic circuitry, or in computer software, firmware, or hardware, including the structures disclosed in this specification and their structural equivalents, or in combinations of one or more of them. Implementations of the subject matter described in this specification can be implemented as one or more computer program products, i.e., one or more modules of computer program instructions encoded on a tangible and non-transitory computer readable medium for execution by, or to control the operation of, data processing apparatus. The computer readable medium can be a machine-

readable storage device, a machine-readable storage substrate, a memory device, a composition of matter effecting a machine-readable propagated signal, or a combination of one or more of them. The term “data processing unit” or “data processing apparatus” encompasses all apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can include, in addition to hardware, code that creates an execution environment for the computer program in question, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them.

[00309] A computer program (also known as a program, software, software application, script, or code) can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment. A computer program does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data (e.g., one or more scripts stored in a markup language document), in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, sub programs, or portions of code). A computer program can be deployed to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a communication network.

[00310] The processes and logic flows described in this specification can be performed by one or more programmable processors executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by, and apparatus can also be implemented as, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit).

[00311] Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a processor for performing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from

or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Computer readable media suitable for storing computer program instructions and data include all forms of nonvolatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[00312] It is intended that the specification, together with the drawings, be considered exemplary only, where exemplary means an example. As used herein, the use of “or” is intended to include “and/or”, unless the context clearly indicates otherwise.

[00313] While this patent document contains many specifics, these should not be construed as limitations on the scope of any invention or of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments of particular inventions. Certain features that are described in this patent document in the context of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially claimed as such, one or more features from a claimed combination can in some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

[00314] Similarly, while operations are depicted in the drawings in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. Moreover, the separation of various system components in the embodiments described in this patent document should not be understood as requiring such separation in all embodiments.

[00315] Only a few implementations and examples are described and other implementations, enhancements and variations can be made based on what is described and illustrated in this patent document.

CLAIMS

1. A method for video processing, comprising:
determining, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, an order of coding side information of additional intra coding methods associated with the video processing unit, wherein the order is changed from one video processing unit to another video processing unit; and
performing the conversion based on the determined order.
2. The method of claim 1, wherein the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method.
3. The method of claim 2, wherein the additional intra coding methods include at least one of an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode or an intra subblock partitioning (ISP) mode.
4. The method of any of claims 1-3, wherein the video processing unit is one of a sequence, view, picture, slice, tile, brick, coding tree unit (CTU) row, largest coding unit (LCU), Virtual Pipelining Data Units (VPDU), coding unit (CU), prediction unit (PU) and transform unit (TU).
5. The method of any of claims 1-4, wherein an indication of the order is signaled in at least one of a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a slice header, a tile header, a coding tree unit (CTU) row, a largest coding unit (LCU) or a virtual pipelining data unit (VPDU).
6. The method of any of claims 1-5, wherein the order depends on dimensions of the video processing unit.
7. The method of any of claims 1-6, wherein the order is based on coded information of one or multiple previous coded video processing units.

8. The method of claim 7, wherein a history table is maintained and updated which records prediction methods of the multiple previous coded video processing units which are the most-recent intra-coded video processing units.
9. The method of claim 7, wherein the order depends on occurrence and/or frequency of each coding mode in the previously coded video processing units.
10. The method of claim 7, wherein the order depends on coded methods of spatial adjacent or non-adjacent and/or temporal neighboring coded video processing units of the video processing unit.
11. The method of any of claims 1-10, wherein the order depends on whether one or multiple intra coding methods are applicable for the video processing unit.
12. The method of claim 11, wherein for certain dimension of the video processing unit, certain method is disabled, and the side information of the certain method is not signaled.
13. A method for video processing, comprising:
coding, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, an indication of usage of additional intra coding methods or conventional intra prediction method for the video processing unit in the bitstream, wherein the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method, and the conventional intra prediction method includes one or more intra coding methods that use adjacent line or column for intra prediction which uses interpolation filter along prediction direction; and
performing the conversion based on the indication.
14. The method of claim 13, wherein the additional intra coding methods includes at least one of an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode or an intra subblock partitioning (ISP) mode.

15. The method of any of claims 13-14, wherein the conventional intra prediction method includes a wide-angle intra-prediction method.
16. The method of any of claims 13-14, wherein the conventional intra prediction method does not include a wide-angle intra-prediction method.
17. The method of any of claims 13-16, wherein a one-bit flag is coded to indicate the conventional intra prediction method is used or not.
18. The method of any of claims 13-17, wherein a pulse code modulation (PCM) mode flag is coded before the indication, where the PCM mode flag is a one-bit flag.
19. The method of any of claims 13-18, wherein a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode flag is coded before the indication, where the QR-BDPCM mode flag is a one-bit flag.
20. The method of any of claims 13-19, wherein the additional intra coding methods include a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.
21. The method of any of claims 13-19, wherein the additional intra coding methods exclude a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.
22. The method of any of claims 13-21, wherein syntax elements related to one or multiple of the additional intra coding methods are coded after the indication of the conventional intra prediction mode.
23. The method of any of claims 13-22, wherein a one-bit flag is coded to indicate whether the conventional intra prediction method or the additional intra coding methods is used.

24. The method of claim 23, wherein if the flag indicates the conventional intra prediction method is used, the signaling of the additional intra coding methods is skipped, and all the additional intra coding methods are inferred to be disabled.
25. The method of claim 23, wherein if the flag indicates the additional intra prediction methods are used, syntax elements related to the additional intra coding methods are further coded in a given order.
26. The method of claim 25, wherein the order is defined as ALWIP mode, MRL mode and ISP mode.
27. The method of claim 25, wherein any order of ALWIP mode, MRL mode and ISP mode is utilized as the given order.
28. The method of claim 25, wherein the order is defined as QR-BDPCM mode, ALWIP mode, MRL mode and ISP mode.
29. The method of claim 25, wherein any order of QR-BDPCM mode, ALWIP mode, MRL mode and ISP mode is utilized as the given order.
30. The method of claim 25, wherein the order is defined as ALWIP mode, MRL mode, ISP mode and other new coding methods.
31. The method of claim 25, wherein the order is adaptively changed.
32. The method of any of claims 25-31, wherein signaling of usage of the last intra coding method is skipped.
33. The method of claim 32, wherein if the order is ALWIP mode, MRL mode and ISP mode, decoded syntax elements indicate ALWIP mode and MRL mode are both disabled, signaling of the usage of ISP mode indicated by `intra_subpartitions_mode_flag` is skipped.

34. The method of claim 32, wherein if the signaling is skipped, the usage of the last intra coding method is inferred.
35. The method of claim 23, wherein the one-bit flag is context coded with one or multiple contexts in arithmetic coding.
36. The method of claim 35, wherein when multiple contexts are allocated, one of the multiple contexts is selected according to information of neighboring video processing units including information of whether the neighboring video processing unit is coded with the conventional intra prediction method.
37. The method of claim 35, wherein the one or multiple contexts is selected based on bin index of a bin.
38. The method of claim 23, wherein the one-bit flag is conditionally signaled.
39. The method of claim 38, wherein when all of the additional intra coding methods are disabled in at least one of one slice, tile group, tile, brick, picture and sequence, the one-bit flag is not coded.
40. The method of claim 38, wherein the one-bit flag is not coded according to dimension of the video processing unit.
41. The method of any of claims 13-40, wherein whether to follow prior design or the coding of the indication is signaled in at least one of sequence, picture, slice, tile group, tile, brick, CU and video data unit level.
42. The method of claim 41, wherein whether to enable the coding of the indication is signaled in at least one of a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a

sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs).

43. A method for video processing, comprising:
constructing, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, a most probable mode (MPM) list by unifying a MPM list construction process for all additional intra coding methods associated with the video processing unit; and
performing the conversion based on the MPM list.
44. The method of claim 43, wherein the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method.
45. The method of claim 43 or 44, wherein the additional intra coding methods include an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode and an intra subblock partitioning (ISP) mode.
46. The method of claim 43 or 44, wherein the additional intra coding methods include an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode, an intra subblock partitioning (ISP) mode, and a quantized residual block differential pulse-code modulation (QR-BDPCM) coding mode.
47. The method of any of claims 43-46, wherein an additional stage is invoked to derive modified intra prediction mode candidates for one additional intra coding method.
48. The method of claim 47, wherein the additional stage is invoked to reorder the MPM list, or discard a mode, or replacing one mode in the MPM list by another one.
49. The method of any of claims 43-48, wherein for each coding methods, a given set of allowed intra prediction modes is pre-defined and/or signaled.

50. The method of claim 49, wherein modes in the MPM list are further mapped to one of allowed intra prediction modes in the given set for one additional intra coding method.
51. The method of any of claims 47-50, wherein different intra coding method selects partial or full of modes in the MPM list.
52. The method of any of claim 43-51, wherein one or multiple of the additional intra coding methods are allowed only for certain intra-prediction modes.
53. The method of claim 52, wherein the one or more of the additional intra coding methods include MRL mode, or/and ISP mode or/and QR-BDPCM mode.
54. The method of any of claim 52-53, wherein one or multiple of the additional intra coding methods are allowed for K intra-prediction modes in the MPM list, K being an integer.
55. The method of claim 54, wherein K is 2 or 3.
56. The method of claim 54 or 55, wherein the K intra-prediction modes are the first K intra-prediction mode in the MPM list.
57. The method of any of claims 52-53, wherein one or multiple of the additional intra coding methods are allowed for pre-defined sets of intra-prediction modes including Planar, DC, vertical, horizontal mode.
58. The method of any of claims 52-53, wherein a first K intra modes in the MPM list are checked in order, and when an intra mode in the MPM list is also part of a predefined or signaled intra-prediction mode set, the one or multiple of the additional intra-prediction coding methods are allowed for the intra mode, K being an integer.
59. The method of any of claims 52-53, wherein intra-prediction modes in the MPM list are checked in order until K valid intra-prediction modes are found or all modes in the MPM list are

checked, wherein when the intra-prediction mode is included in a predefined intra mode set, the intra-prediction mode is considered as valid.

60. A method for video processing, comprising:
signaling, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, intra-prediction modes information before usage of one or multiple additional intra coding methods associated with the video processing unit; and performing the conversion based on the intra-prediction modes information.
61. The method of claim 60, wherein the additional intra coding methods include one or more intra coding methods that require additional signaling for the usage for the method.
62. The method of claim 60 or 61, wherein the one or multiple additional intra coding methods include a multiple reference line (MRL) intra prediction mode and/or an intra subblock partitioning (ISP) mode.
63. The method of any of claims 60-62, wherein the usage of one or multiple additional intra coding methods is conditionally signaled according to the intra-prediction mode information.
64. The method of any of claims 60-63, wherein if decoded intra-prediction mode is not allowed for one additional intra coding method, the usage of the one additional intra coding method is not signaled any more.
65. The method of any of claims 60-64, wherein all of the additional intra coding methods including MRL mode, ISP mode and ALWIP mode are coded after the signaling of intra-prediction modes including `intra_luma_mpm_flag`, `intra_luma_not_planar_flag`, `intra_luma_mpm_idx` or `intra_luma_mpm_remainder`.
66. The method of any of claims 60-64, wherein partial of the additional intra coding methods including MRL mode and ISP mode are coded after the signaling of intra-prediction modes including `intra_luma_mpm_flag`, `intra_luma_not_planar_flag`, `intra_luma_mpm_idx` or

intra_luma_mpm_remainder, and remaining of the additional intra coding methods including QR-BDPCM mode and/or ALWIP mode is coded before the signaling of intra prediction modes.

67. The method of any of claims 60-64, wherein indications of usage of the additional intra coding methods including QR-BDPCM mode, MRL mode and ALWIP mode is coded before the signaling of intra-prediction modes, while ISP mode is coded after the signaling of intra prediction modes.

68. The method of any of claims 63-64, wherein the signaling of usage of the one or multiple additional intra coding methods depends on whether the intra-prediction mode corresponds to wide-angle intra-prediction.

69. A method for video processing, comprising:
 signaling, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, one or multiple additional intra coding methods related syntax elements after certain intra coding information; and
 performing the conversion based on the one or multiple additional intra coding methods related syntax elements.

70. The method of claim 69, wherein the one or multiple additional intra coding methods include a multiple reference line (MRL) intra prediction mode and/or an intra subblock partitioning (ISP) mode and/or other methods.

71. The method of claim 69 or 70, wherein whether signaling the one or multiple additional intra coding methods related syntax elements depends on the intra coding information.

72. The method of any of claims 69-71, wherein the intra coding information includes a MPM flag indicating whether an intra-prediction mode for the video processing unit is from a most probable mode (MPM) list, wherein the MPM flag is denoted by intra_luma_mpm_flag.

73. The method of claim 72, wherein side information of the one or multiple additional intra coding methods is conditional signaled after or right after the MPM flag, wherein the side information includes at least one of `intra_luma_ref_idx`, `intra_subpartitions_mode_flag`, `intra_subpartitions_split_flag`.

74. The method of any of claims 69-73, wherein the intra coding information includes a planar mode flag indicating whether an intra-prediction mode for the video processing unit is planar or not, or a first MPM flag indicating whether the intra-prediction mode is a first MPM candidate in the MPM list or not, wherein the planar mode flag is denoted by `intra_luma_not_planar_flag`.

75. The method of claim 74, wherein side information of the one or multiple additional intra coding methods is conditional signaled after or right after the planar mode flag or the first MPM flag.

76. The method of any of claims 69-75, wherein the intra coding information includes the remaining MPM index which is denoted by `intra_luma_mpm_idx`.

77. The method of claim 76, wherein side information of the one or multiple additional intra coding methods is conditional signaled after or right after the remaining MPM index.

78. The method of any of claims 69-77, wherein the intra coding information includes intra-prediction mode which is indicated by at least one of `intra_luma_not_planar_flag`, `intra_luma_mpm_idx` and `intra_luma_mpm_remainder`.

79. The method of claim 78, wherein side information of the one or multiple additional intra coding methods is conditional signaled after or right after the intra-prediction mode.

80. The method of claim 78, wherein when intra-prediction mode of the block is not from the MPM list with the MPM flag `intra_luma_mpm_flag` is equal to 0, the one or multiple additional intra coding methods related syntax elements are not signaled.

81. The method of claim 80, wherein the signaling of the MPM flag `intra_luma_mpm_flag` is independent from usages of ISP mode and MRL mode.
82. The method of claim 81, wherein the MPM flag `intra_luma_mpm_flag` is signaled without conditional check of `intra_subpartitions_mode_flag` and `intra_luma_ref_idx` are both equal to 0.
83. The method of claim 80, wherein when one method related syntax is not signaled, the method is not applied to the video processing unit.
84. The method of claim 83, wherein when `intra_subpartitions_mode_flag` is inferred to be 0, ISP mode is not applied for the video processing unit.
85. The method of claim 83, wherein when `intra_luma_ref_idx` is inferred to be 0, MRL mode is not applied for the video processing unit.
86. The method of claim 70, wherein signaling of MRL or/and ISP related syntax elements is further dependent on dimensions of the video processing unit.
87. The method of claim 70, wherein MRL related syntax elements is not signaled for planar mode.
88. The method of claim 87, wherein when `intra_luma_not_planar_flag` is equal to 0, signaling of the usage of MRL is skipped.
89. The method of claim 87, wherein when `intra_luma_ref_idx` is inferred to be 0, MRL mode is not applied for the video processing unit.
90. A method for video processing, comprising:

constructing, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, candidate intra coding method list (IPMList) for the video processing unit; and

performing the conversion based on the IPMList.

91. The method of claim 90, wherein an index IPMI_{idx} to the IPMList is coded.

92. The method of claim 91, wherein IPMI_{idx} associated with spatial neighboring video processing units is inserted to the IPMList in order, where the spatial neighboring video processing units include spatial neighboring adjacent or non-adjacent video processing units.

93. The method of claim 91 or 92, wherein conventional intra coding method is a first candidate in the IPMList, wherein the conventional intra prediction method includes one or more intra coding methods that use adjacent line or column for intra prediction which uses interpolation filter along prediction direction.

94. The method of any of claim 91-93, wherein the spatial neighboring video processing units are defined to be those which are used in motion candidate list construction process of inter-AMVP code, merge mode, affine mode, IBC mode or the MPM list construction process in the normal intra prediction method.

95. The method of any of claim 91-94, wherein checking order of the spatial neighboring blocks are defined to be the same or different as that used in the motion candidate list construction process of inter-AMVP code, merge mode, affine mode, IBC mode or the MPM list construction process in the normal intra prediction method.

96. The method of any of claim 91-95, wherein the inserted IPMI_{idx} to the list is further refined.

97. The method of any of claim 91-96, wherein IPMI_{idx} is binarized with truncated unary code, k-th order EG code or fixed length code.

98. A method for video processing, comprising:
jointly coding, for a conversion between a video processing unit of a video and a bitstream representation of the video processing unit, indications of multiple intra coding methods associated with the video processing unit by using one syntax element; and
performing the conversion based on the coded indications.
99. The method of claim 98, wherein the one syntax element is used to indicate a selected intra coding method.
100. The method of claim 99, wherein the selected intra coding method includes at least one of conventional intra coding method and additional intra coding methods including an affine linear weighted intra prediction (ALWIP) mode, a multiple reference line (MRL) intra prediction mode and an intra subblock partitioning (ISP) mode.
101. The method of any of claims 98-100, wherein an intra coding method is represented by an index IPMI_{idx} and the corresponding index of one selected coding method for the video processing unit is coded.
102. The method of any of claims 98-101, wherein the syntax element is binarized with truncated unary, k-th order EG or fixed length.
103. The method of any of claims 98-102, wherein bins of a bin string for the syntax element is context-coded or bypass coded.
104. The method of any of claims 98-103, wherein semantics of the syntax element which indicates a mapping of decoded value and the intra prediction method is changed from one video processing unit to another.
105. The method of any of claims 1, 13, 43, 60, 69, 90, 98, wherein whether to enable or disable the determining process, the coding process, the constructing process, the signaling

process or the jointly coding process is signaled in at least one of a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs).

106. The method of any of claims 1, 13, 43, 60, 69, 90, 98, wherein which process of the determining process, the coding process, the constructing process, the signaling process or the jointly coding process is to be used is signaled in at least one of a decoder parameter set (DPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptive parameter set (APS), a video parameter set (VPS), a sequence header, a picture header, a slice header, a tile group header, a tile or a group of coding tree units (CTUs).

107. The method of claim 105 or 106, wherein whether to enable or disable the processes and/or which process is to be used depends on at least one of dimension of the video processing unit, Virtual Pipelining Data Units (VPDU), picture type and low delay check flag.

108. The method of claim 105 or 106, wherein whether to enable or disable the processes and/or which process is to be used depends on at least one of color component and color format.

109. The method of any of claim 1-108, wherein the conversion generates the video processing unit of video from the bitstream representation.

110. The method of anyone of claims 1-108, wherein the conversion generates the bitstream representation from the video processing unit of video.

111. An apparatus in a video system comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to implement the method in any one of claims 1 to 110.

112. A computer program product stored on a non-transitory computer readable media, the computer program product including program code for carrying out the method in any one of claims 1 to 110.

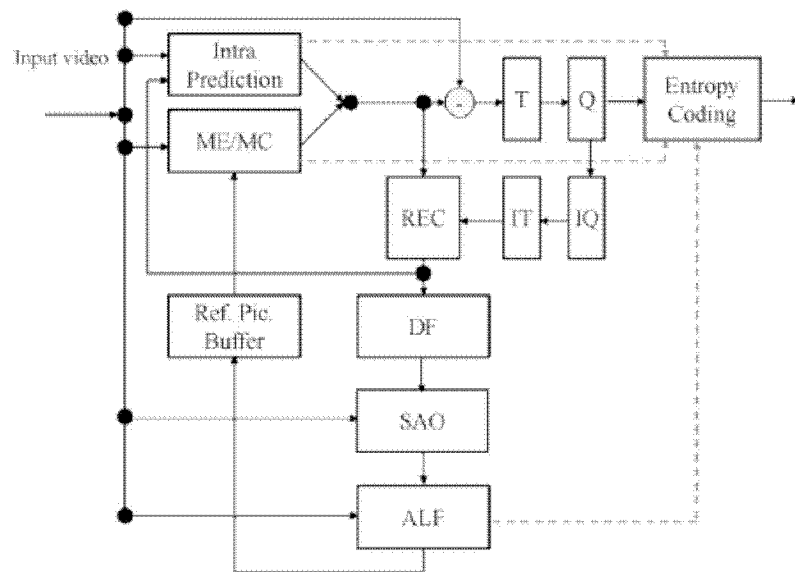


FIG. 1

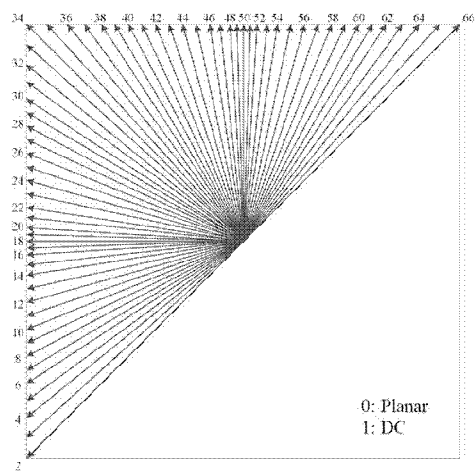


FIG. 2

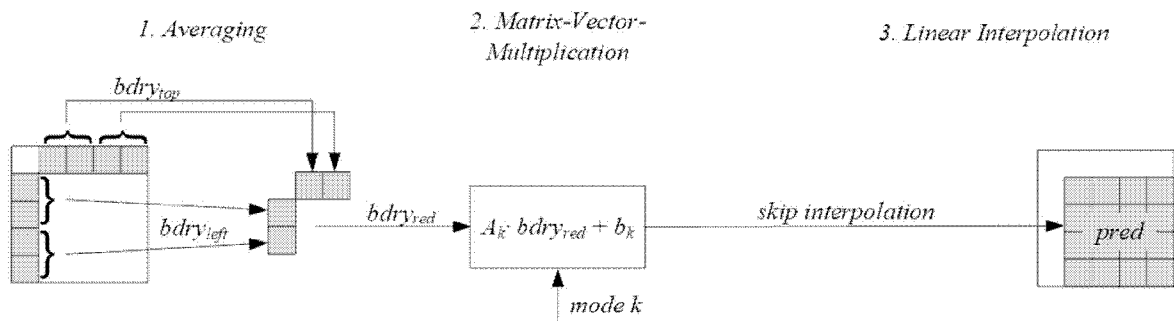


FIG. 3

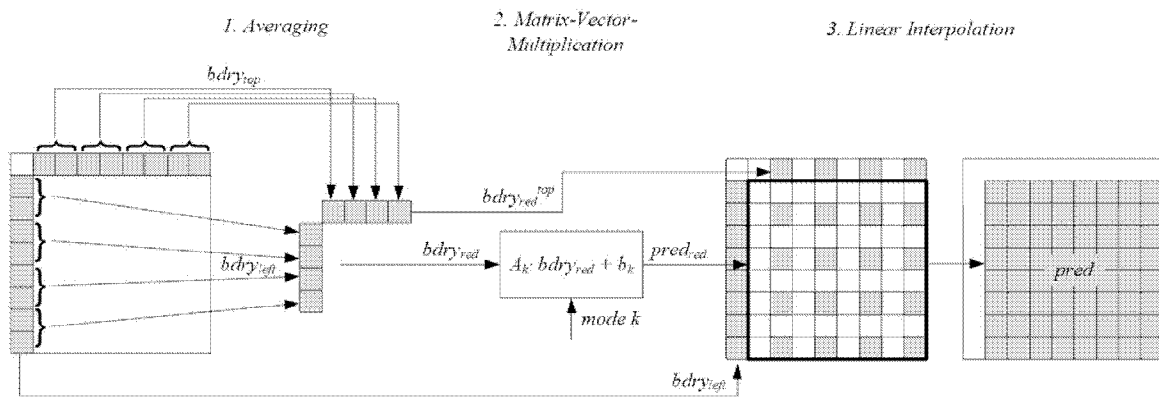


FIG. 4

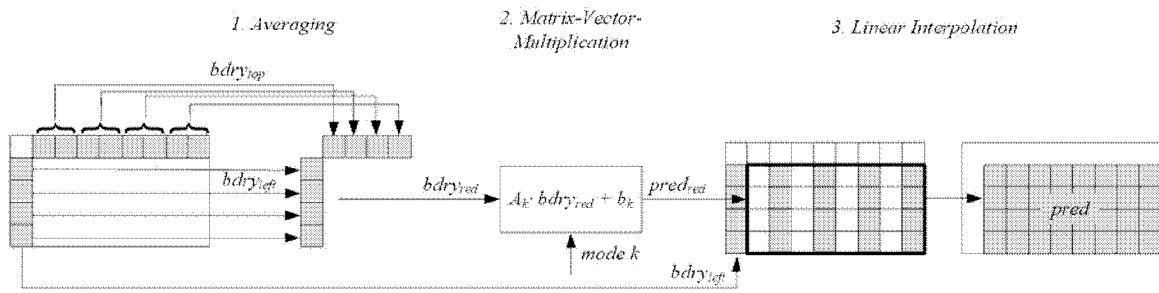


FIG. 5

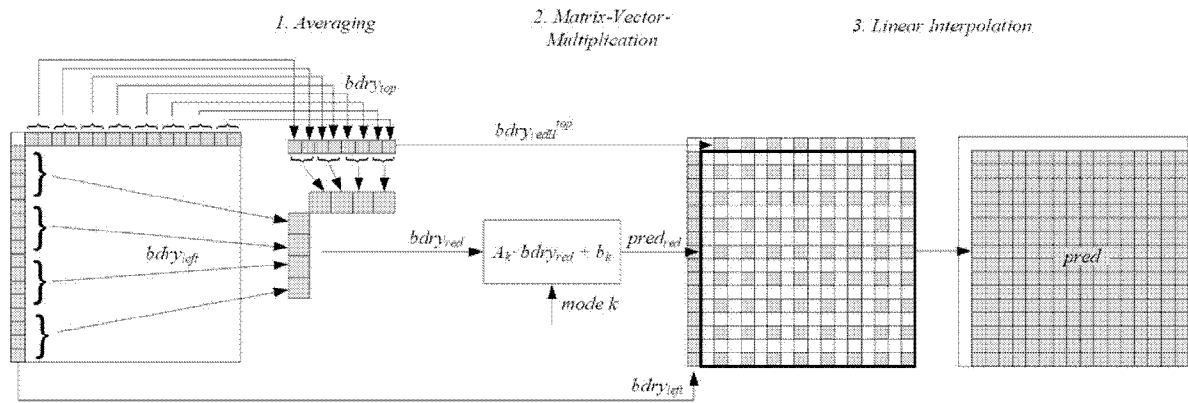


FIG. 6

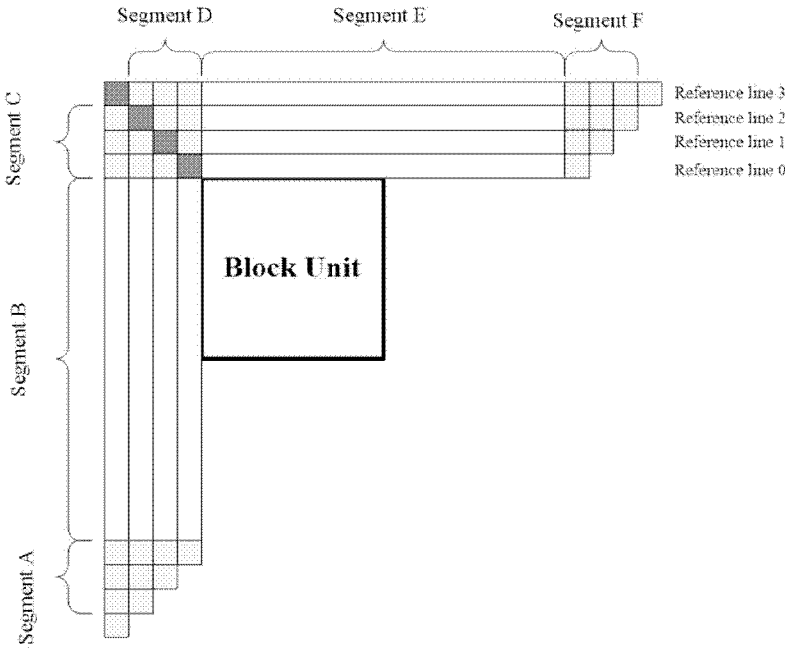


FIG. 7

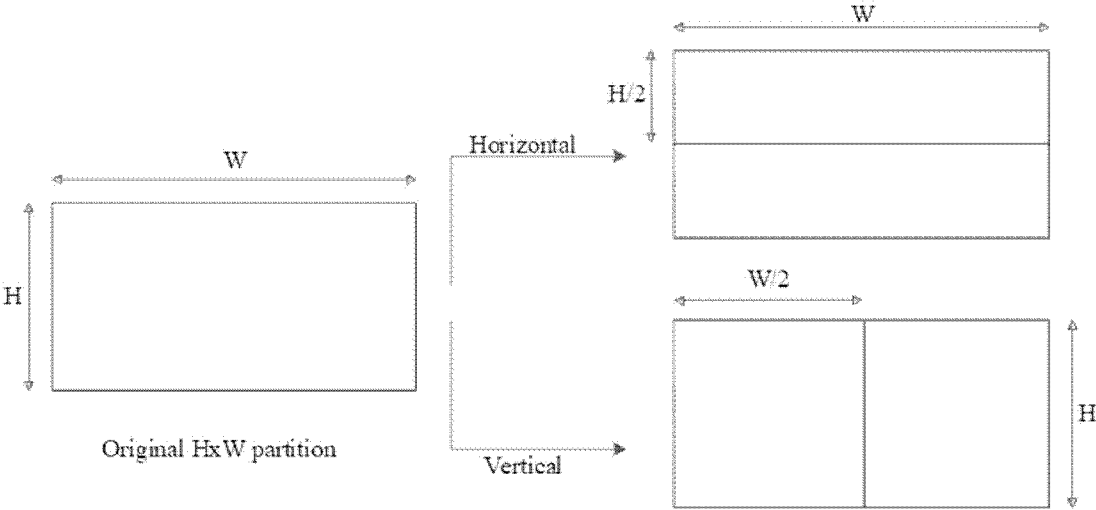


FIG. 8

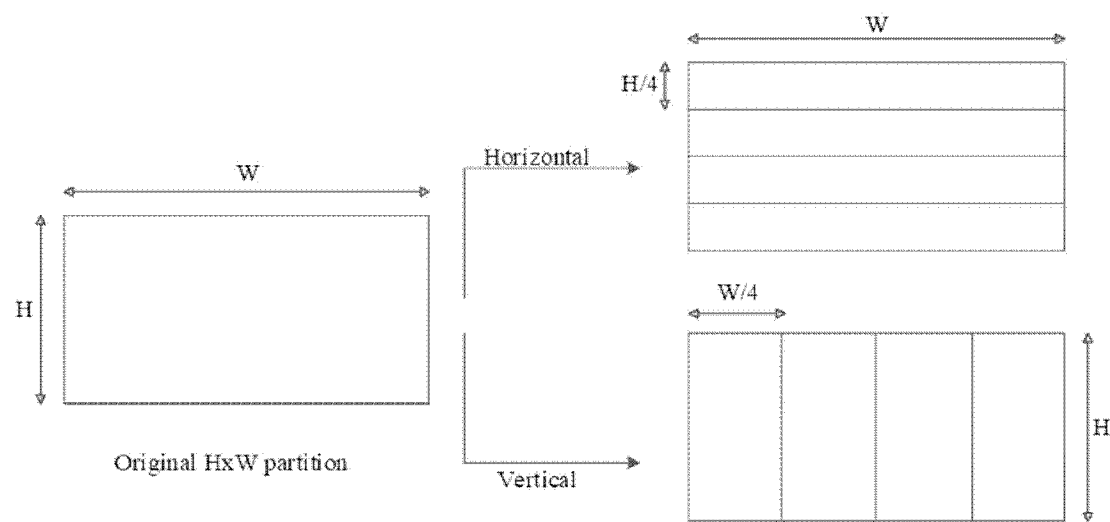


FIG. 9

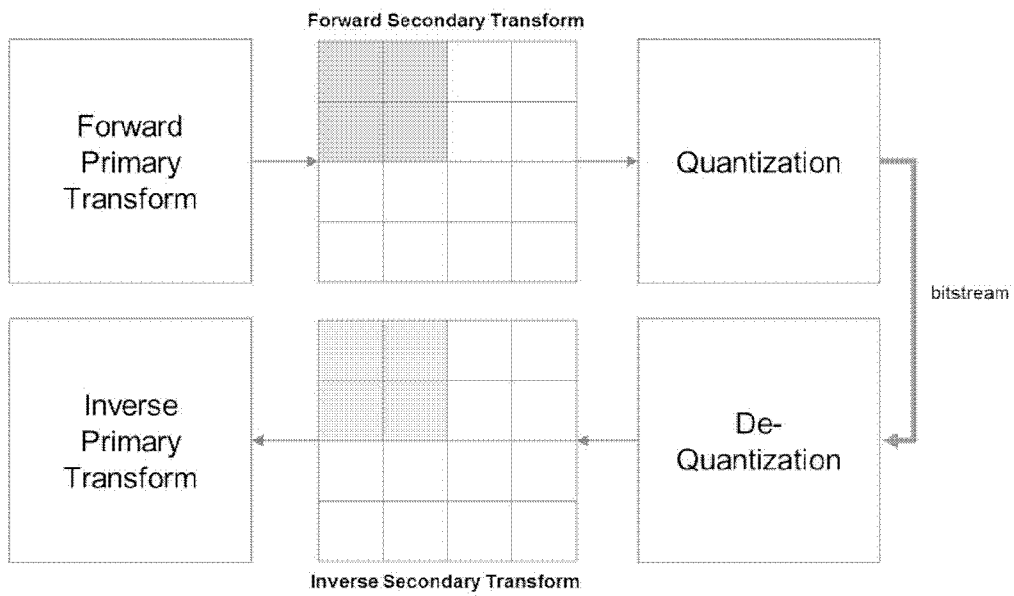


FIG. 10

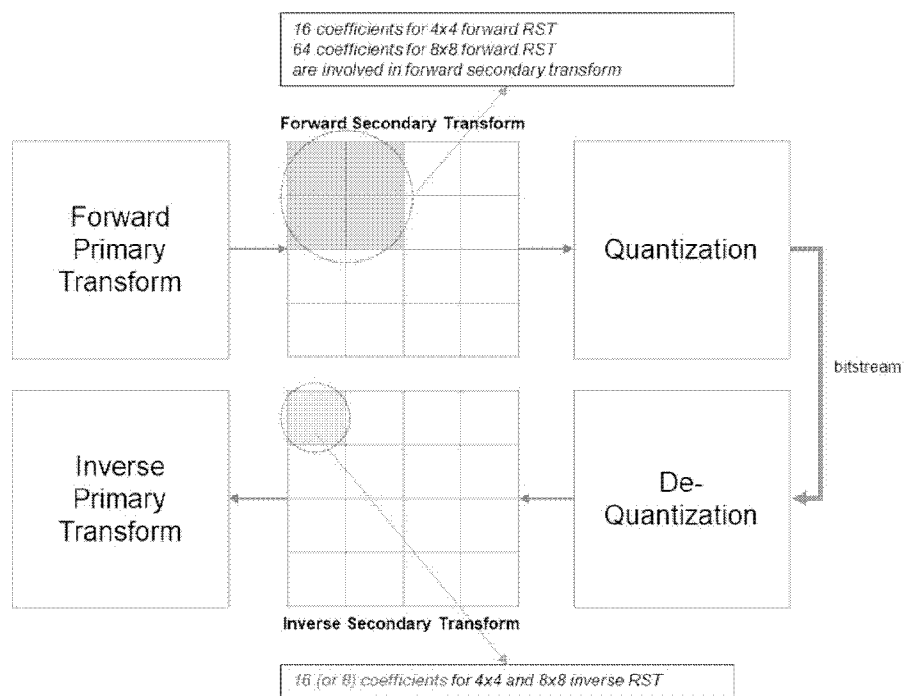


FIG. 11

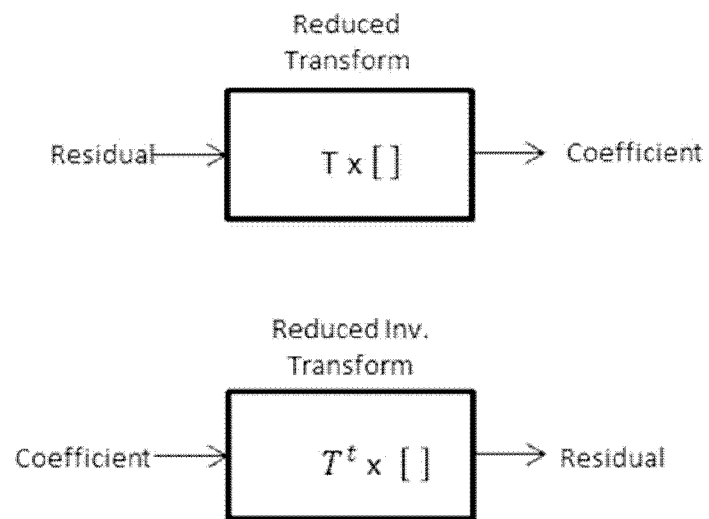


FIG. 12

When $N \geq 8$ and $M \geq 8$,

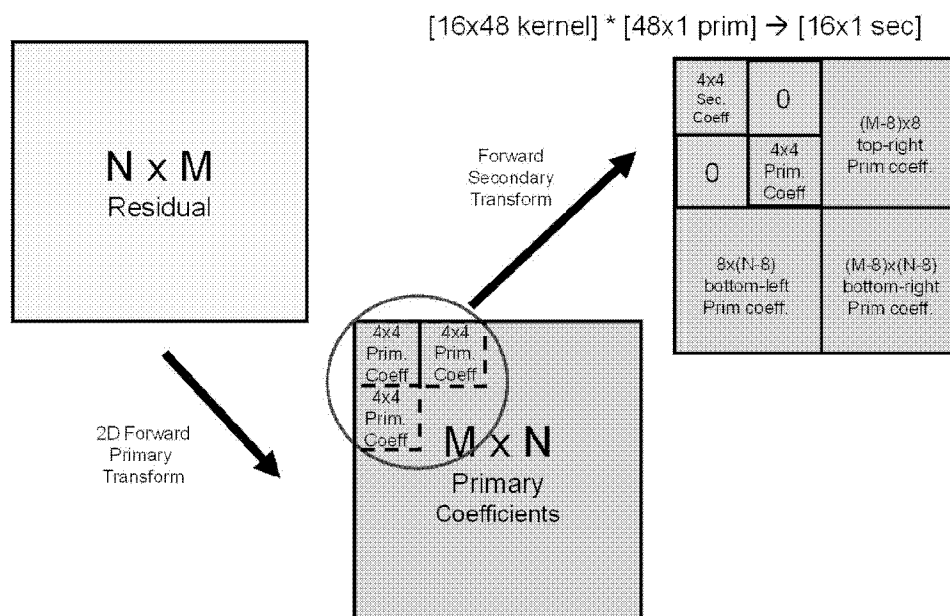


FIG. 13

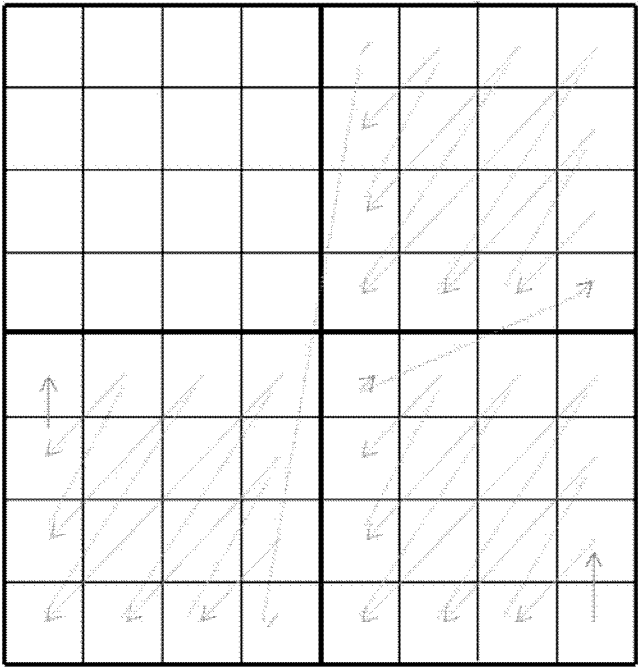


FIG. 14

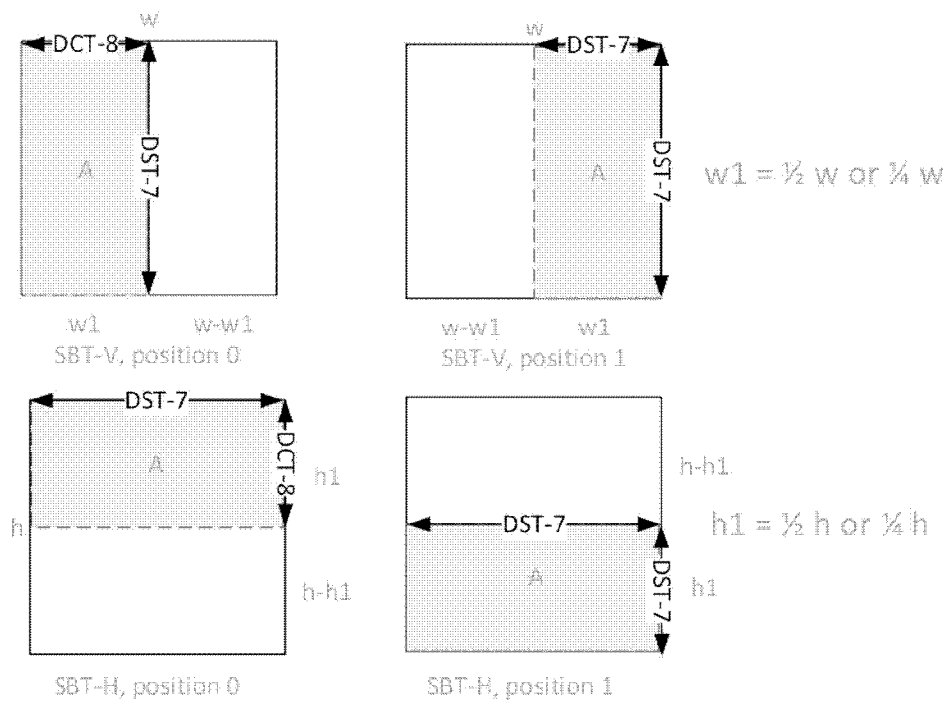


FIG. 15

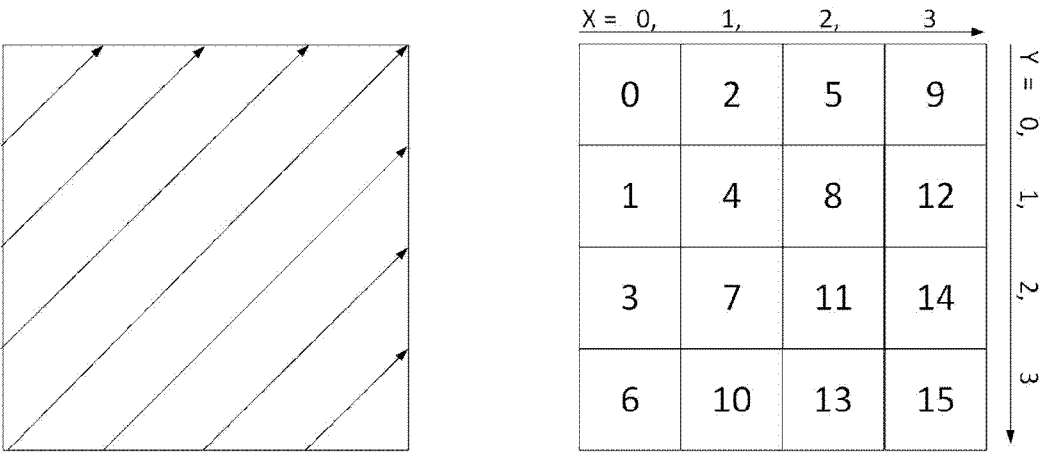


FIG. 16

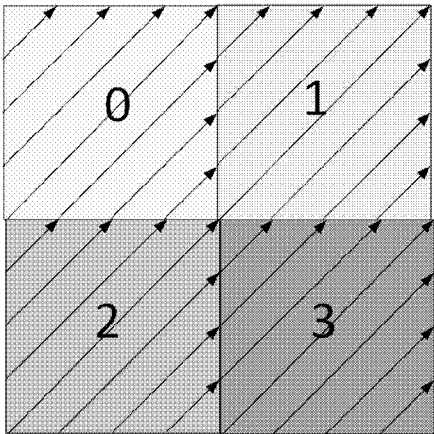


FIG. 17

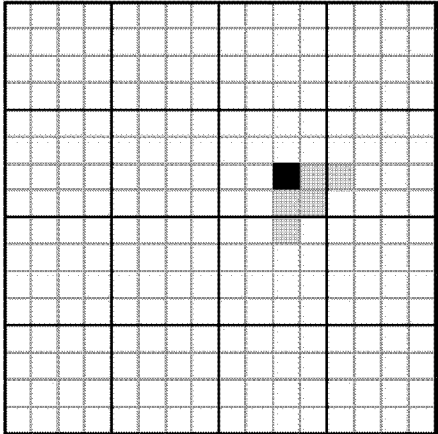


FIG. 18

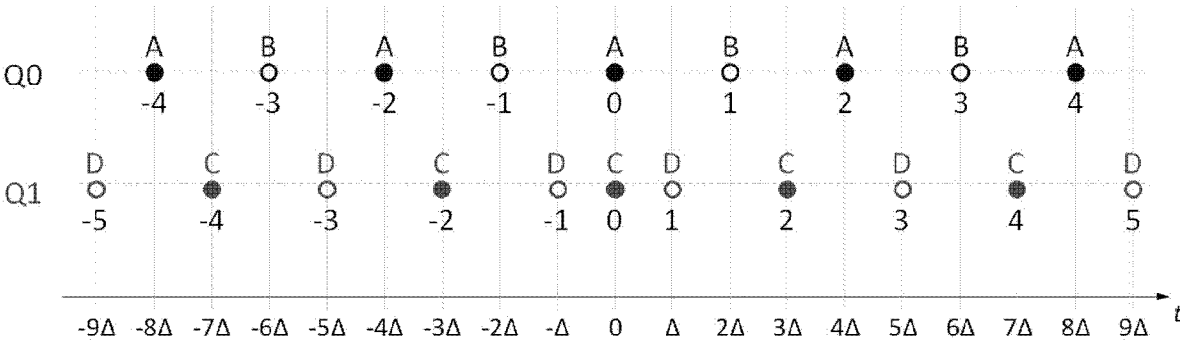
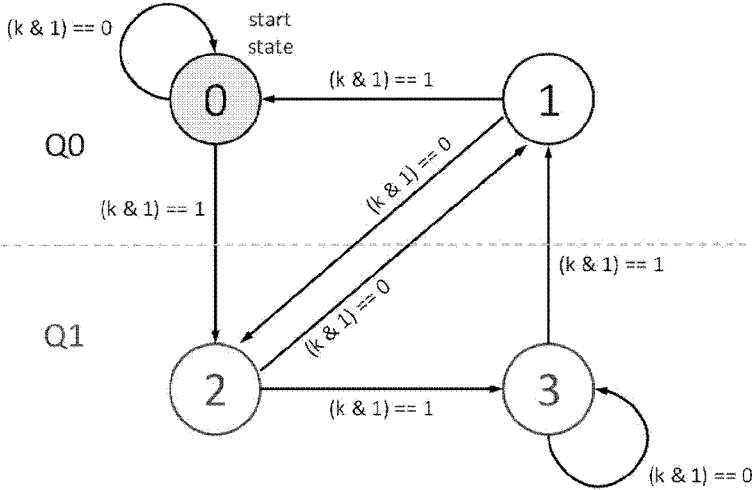


FIG. 19



current state	next state for ...	
	(k & 1) == 0	(k & 1) == 1
0	0	2
1	2	0
2	1	3
3	3	1

FIG. 20

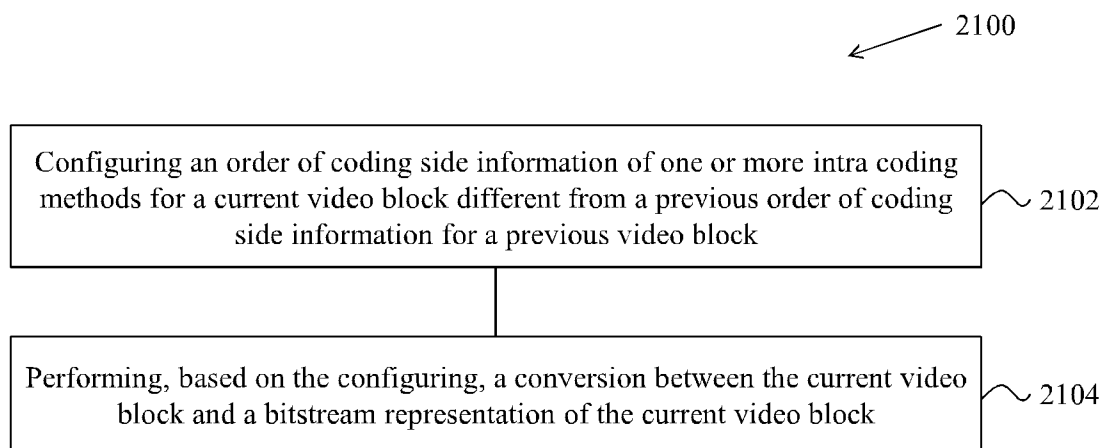


FIG. 21A

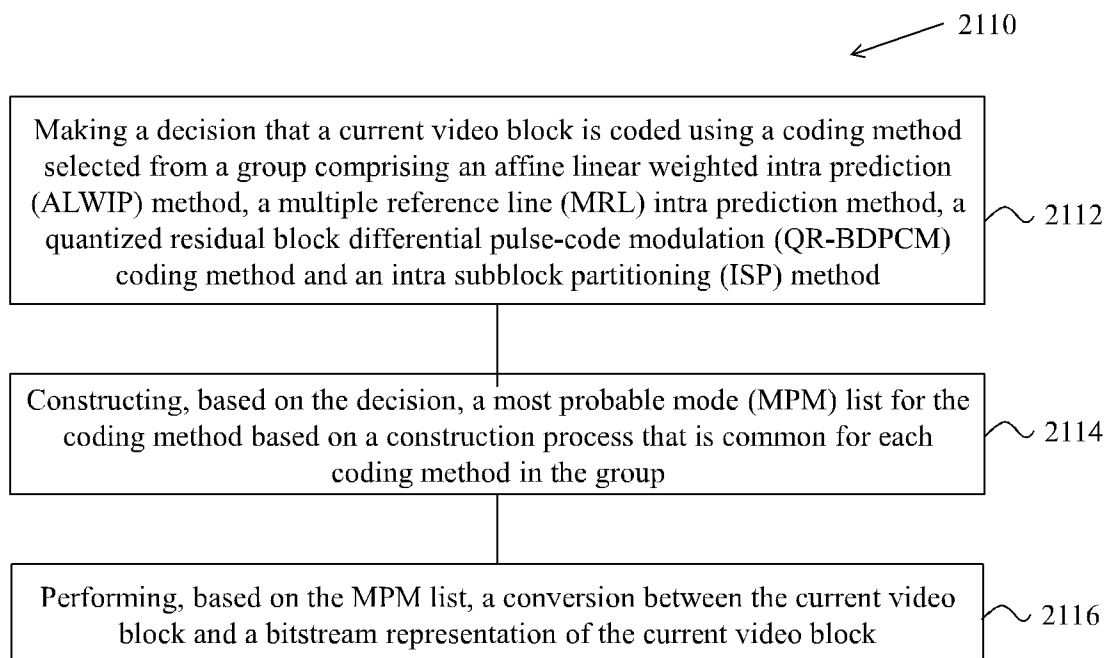


FIG. 21B

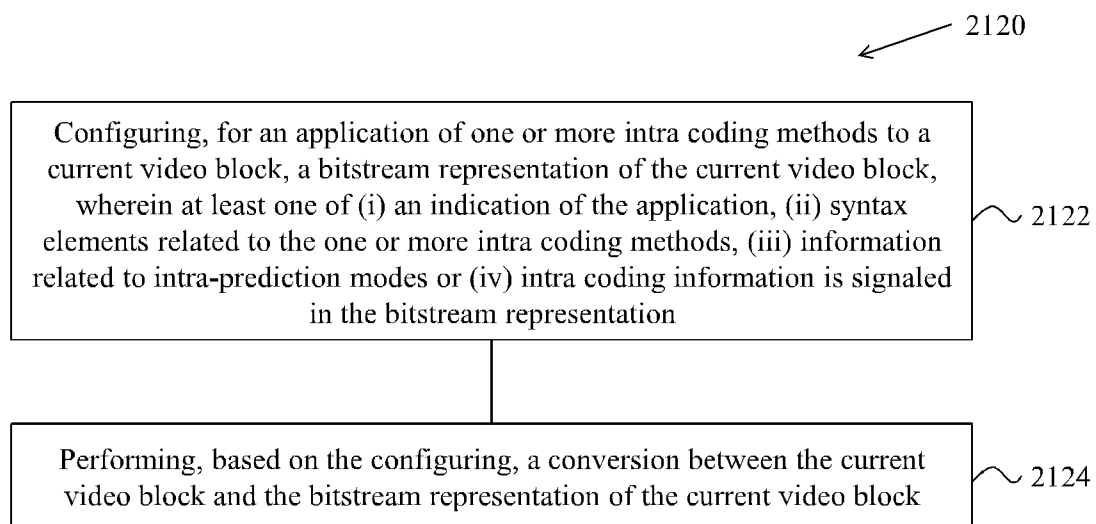


FIG. 21C

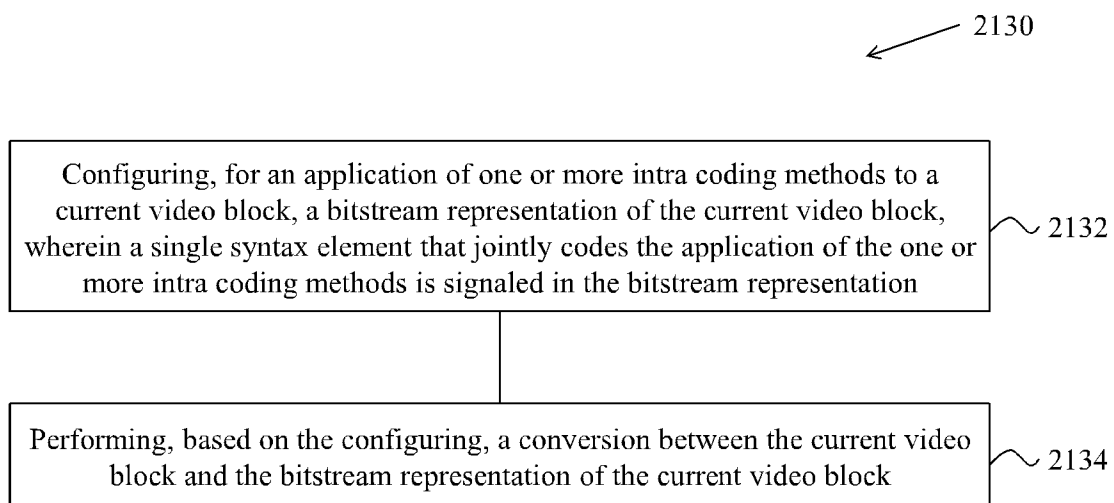


FIG. 21D

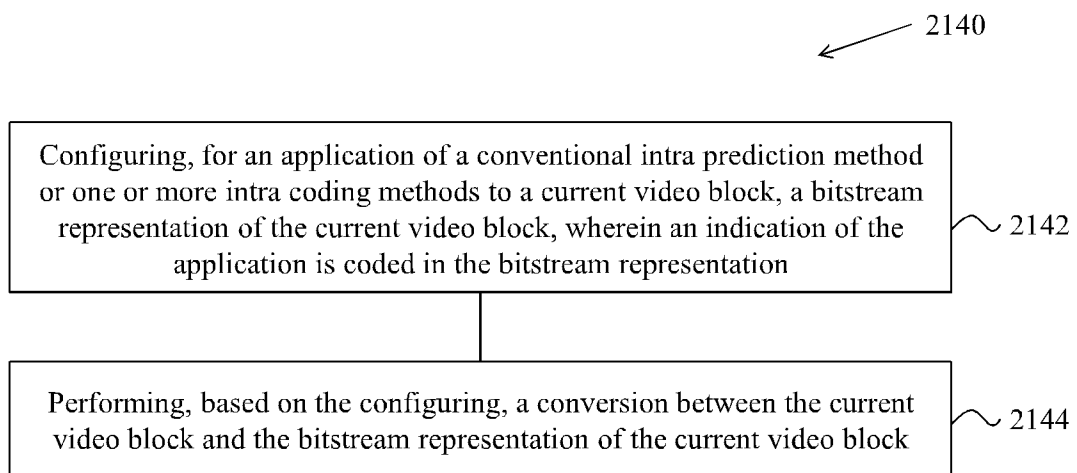


FIG. 21E

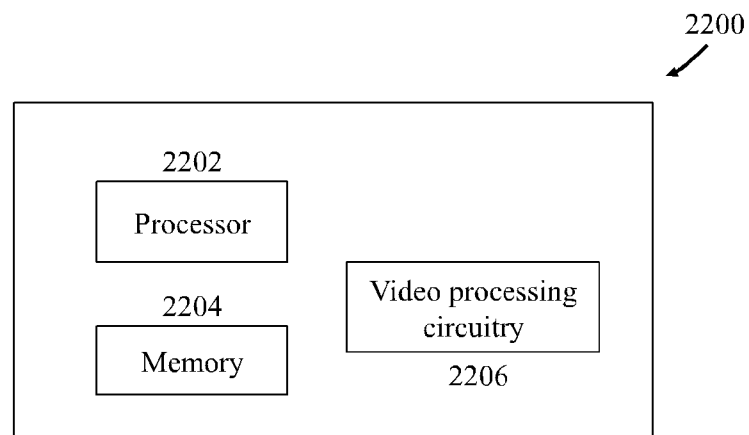


FIG. 22

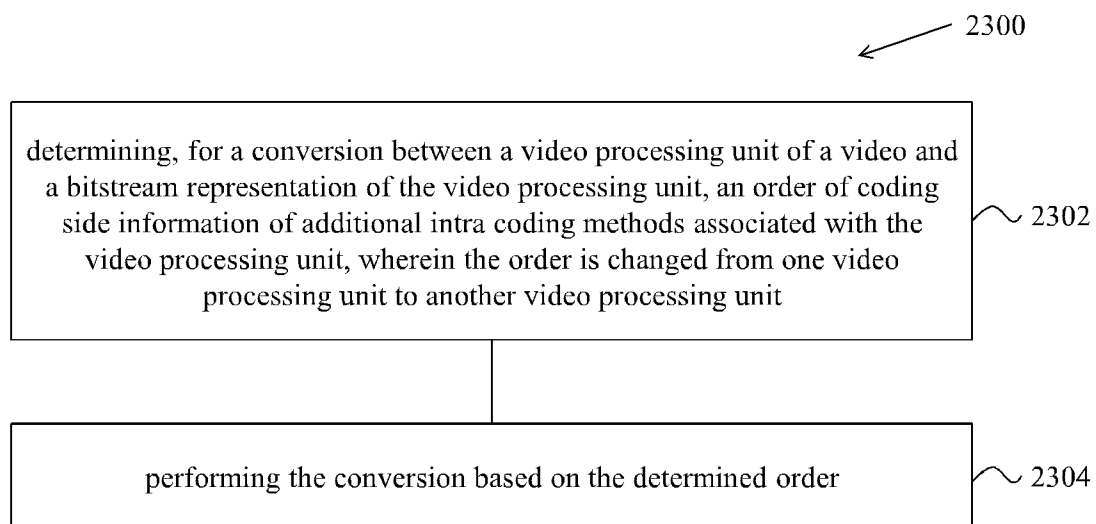


FIG. 23

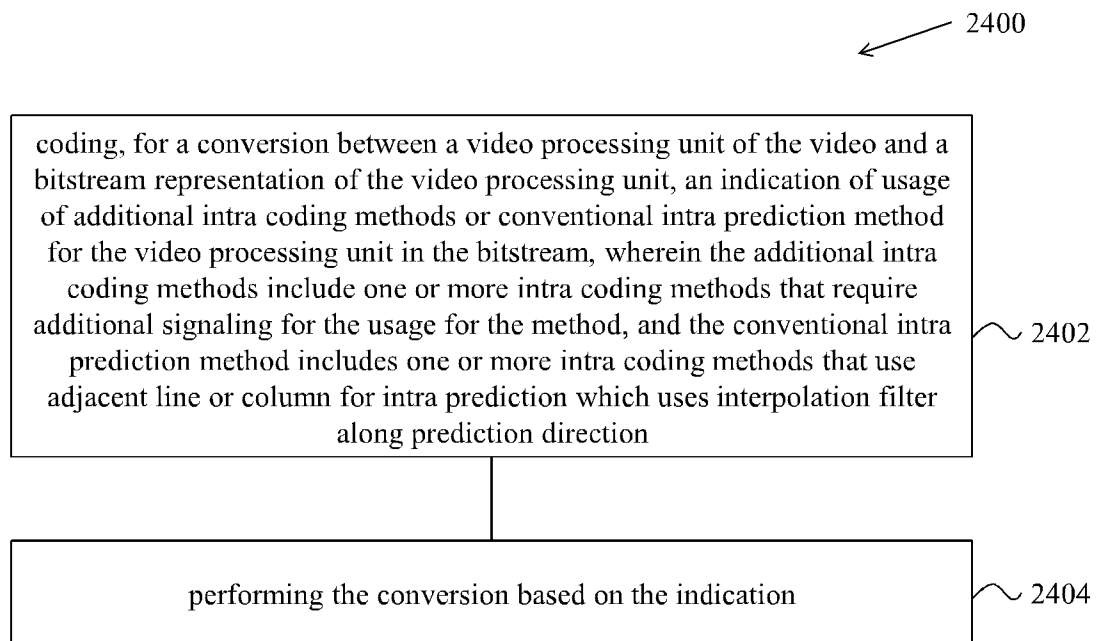


FIG. 24

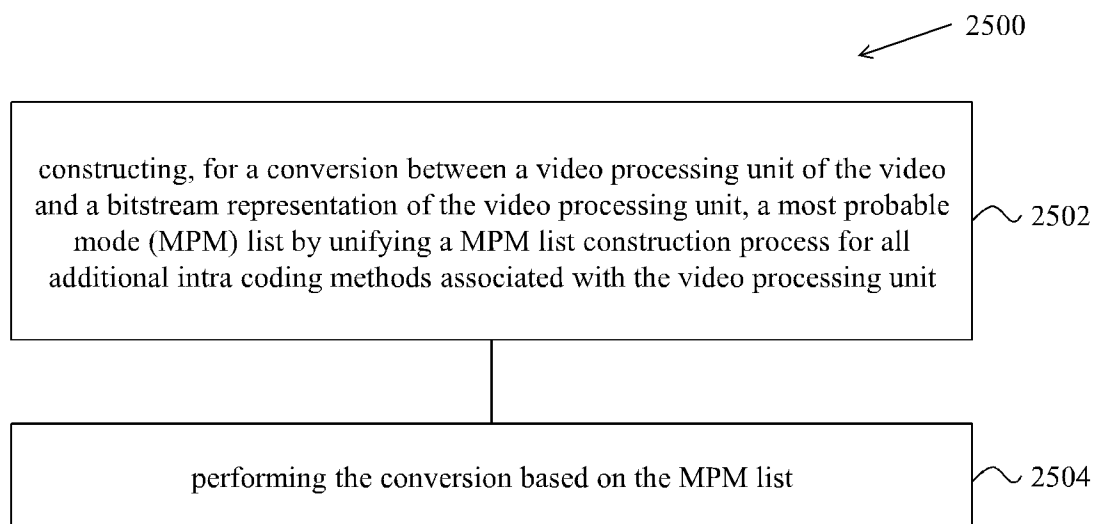


FIG. 25

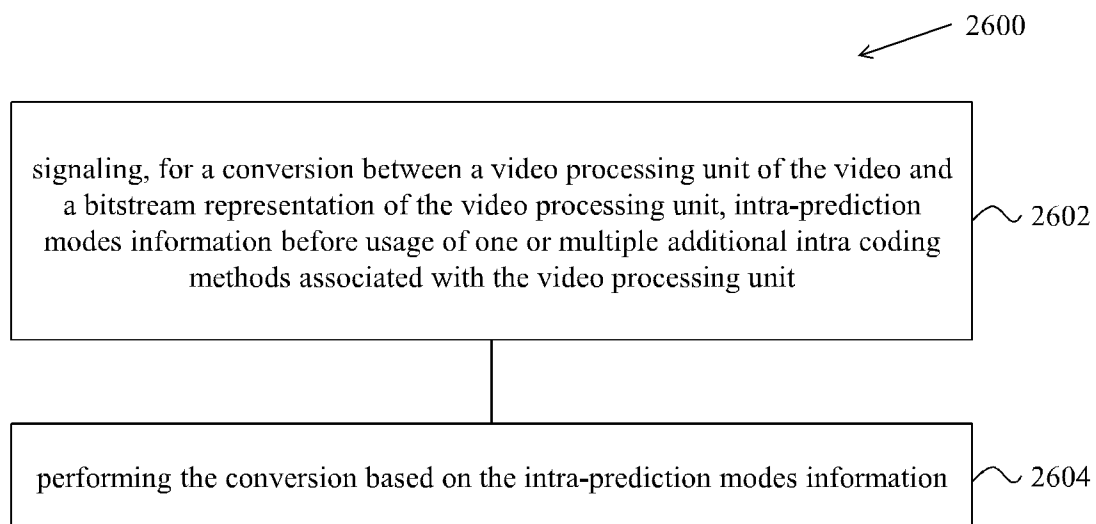


FIG. 26

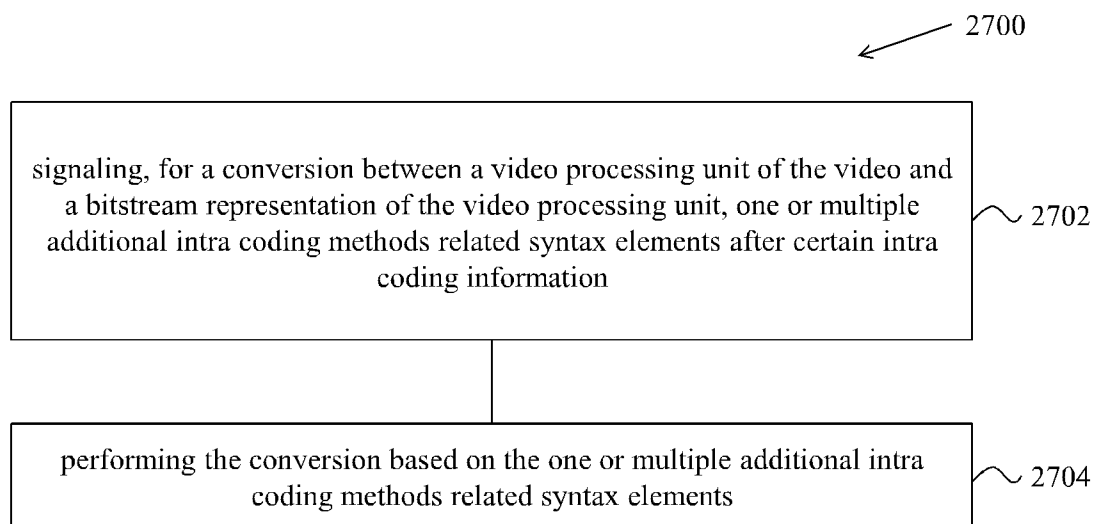


FIG. 27

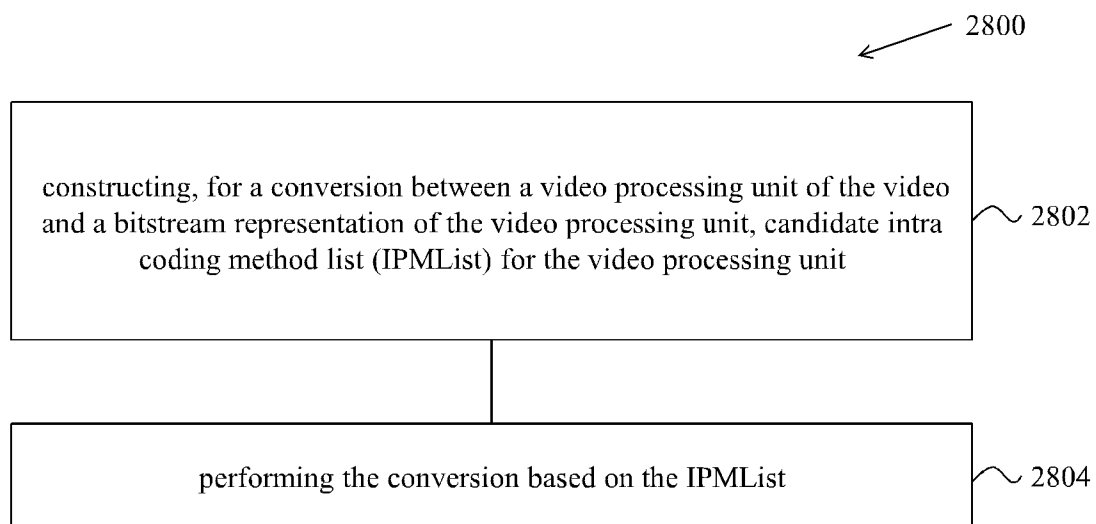


FIG. 28

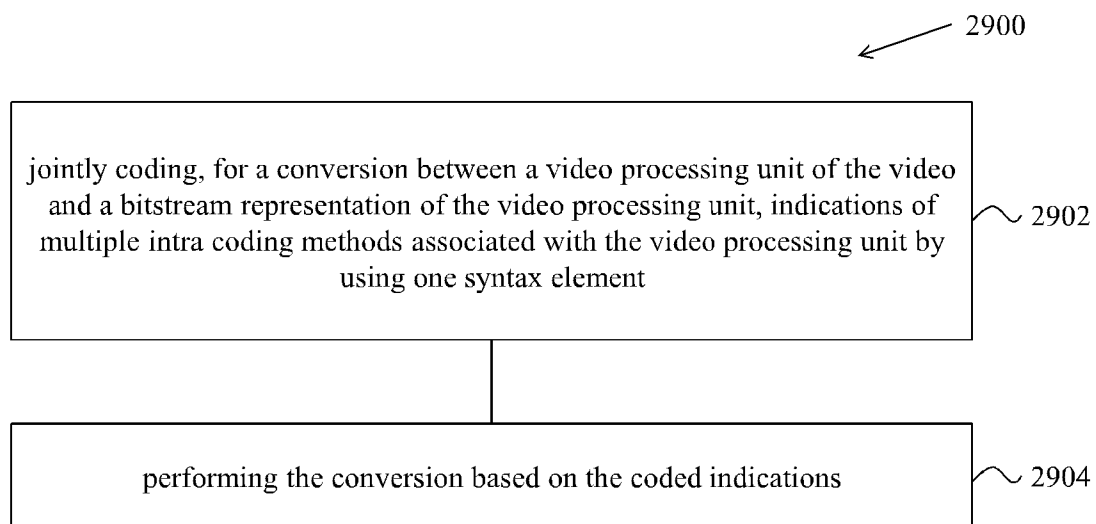


FIG. 29

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/089742

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/52(2014.01)i; H04N 19/567(2014.01)i; H04N 19/537(2014.01)i

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04N

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

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

CNPAT, CNKI, WPI, EPODOC, IEEE: video, image, +cod+, +compress+, side, information, addition+, intra, predict+, order, sequence, ALWIP, MRL, ISP, affine, linear, weighted, multiple, reference, line, subblock, partition+, mode, bytedance

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	PFAFF, Jonathan et al. "CE3: Affine linear weighted intra prediction (test 1.2.1, test 1.2.2) (JVET-M0043)" <i>Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 13th Meeting: Marrakech, MA, 09 January 2019 (2019-01-09),</i> D1 abstract, 1.1, 2.1, 2.5	1-12, 105-112
A	PFAFF, Jonathan et al. "CE3: Affine linear weighted intra prediction (CE3-4.1, CE3-4.2) (JVET-N0217)" <i>Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 14th Meeting: Geneva, CH, 25 March 2019 (2019-03-25),</i> the whole document	1-12, 105-112
A	AUWERA, Geert Van der et al. "CE3: Summary Report on Intra Prediction and Mode Coding (JVET-M0023)" <i>Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 13th Meeting: Marrakech, MA, 09 January 2019 (2019-01-09),</i> the whole document	1-12, 105-112



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search

10 July 2020

Date of mailing of the international search report

29 July 2020

Name and mailing address of the ISA/CN

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

Facsimile No. (86-10)62019451

Authorized officer

XIE,Jiani

Telephone No. 86-(10)-53961703

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/089742

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	SANTIAGO, De-Luxan-Hernandez et al. "CE3: Intra Sub-Partitions Coding Mode (Tests 1.1.1 and 1.1.2) (JVET-M0102)" <i>Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 13th Meeting: Marrakech, MA, 17 January 2019 (2019-01-17),</i> the whole document	1-12, 105-112
A	US 2018234699 A1 (SONY CORPORATION) 16 August 2018 (2018-08-16) the whole document	1-12, 105-112
A	US 2018131936 A1 (INTEL CORPORATION) 10 May 2018 (2018-05-10) the whole document	1-12, 105-112
A	WO 2014160880 A1 (QUALCOMM INCORPORATED) 02 October 2014 (2014-10-02) the whole document	1-12, 105-112
A	WO 2008071036 A1 (THOMSON BROADBAND R & D BEIJING CO., LTD.) 19 June 2008 (2008-06-19) the whole document	1-12, 105-112

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

- [1] Claim 1, which refers to a method comprising determining an order of coding side information of additional intra coding methods associated with the video processing unit.
- [2] Claim 13, which refers to a method comprising coding an indication of usage of additional intra coding methods or conventional intra prediction method for the video processing unit in the bitstream.
- [3] Claim 43, which refers to a method comprising constructing a most probable mode (MPM) list by unifying a MPM list construction process for all additional intra coding methods associated with the video processing unit.
- [4] Claim 60, which refers to a method comprising signaling intra-prediction modes information before usage of one or multiple additional intra coding methods associated with the video processing unit.
- [5] Claim 69, which refers to a method comprising signaling one or multiple additional intra coding methods related syntax elements after certain intra coding information.
- [6] Claim 90, which refers to a method comprising constructing candidate intra coding method list (IPMList) for the video processing unit.
- [7] Claim 98, which refers to a method comprising jointly coding indications of multiple intra coding methods associated with the video processing unit by using one syntax element.
- [8] The seven inventions do not have the same or corresponding technical feature. Further, they do not have the same or corresponding special technical feature. Therefore the inventions are not linked as to form a single general inventive concept, as required by PCT Rule 13.1, 13.2 and 13.3.

- 1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
- 2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.
- 3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:
- 4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.: **1-12, 105-112**

Remark on Protest

- ☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.
- ☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.
- ☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2020/089742

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
US	2018234699	A1	16 August 2018	US	10009629	B2	26 June 2018
				US	10154285	B2	11 December 2018
				US	2015103915	A1	16 April 2015
US	2018131936	A1	10 May 2018	WO	2018089146	A1	17 May 2018
				DE	112017005664	T5	22 August 2019
				CN	109891887	A	14 June 2019
WO	2014160880	A1	02 October 2014	JP	6133489	B2	24 May 2017
				KR	101810757	B1	19 December 2017
				US	9369708	B2	14 June 2016
				JP	2016518051	A	20 June 2016
				CN	105103559	B	15 January 2019
				EP	2979453	A1	03 February 2016
				US	2014301454	A1	09 October 2014
				KR	20150135398	A	02 December 2015
				CN	105103559	A	25 November 2015
				KR	20090098825	A	17 September 2009
				EP	2092747	B1	12 August 2015
WO	2008071036	A1	19 June 2008	JP	2010512698	A	22 April 2010
				JP	5232796	B2	10 July 2013
				CN	101601300	A	09 December 2009
				KR	101345090	B1	26 December 2013
				CN	101601300	B	18 July 2012
				US	2010111167	A1	06 May 2010
				EP	2092747	A1	26 August 2009
				US	8477853	B2	02 July 2013
				EP	2092747	A4	22 December 2010