

CONTEXT CODING FOR TRANSFORM SKIP MODE

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application is a national phase of International Patent Application NO. PCT/US2020/028808, filed on April 17, 2020, which claims the priority to and benefits of International Patent Application No. PCT/CN2019/083366, filed on April 19, 2019. The aforementioned patent applications are hereby incorporated by reference in their entireties.

TECHNICAL FIELD

[0002] This document is related to video and image coding and decoding technologies, systems, and devices.

BACKGROUND

[0003] Digital video 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 coefficient coding in transform skip mode for video coding are described. 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 (e.g., Versatile Video Coding (VVC)) or codecs.

[0005] In one exemplary aspect, a method for visual media processing is disclosed. The method includes determining, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, a maximum allowed dimension up to which a current video block of the one or more video blocks is allowed to be encoded using a transform skip mode such that a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and including a syntax element indicative of the maximum allowed dimension in the bitstream representation.

[0006] In another exemplary aspect, a method for visual media processing is disclosed. The method includes parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a maximum allowed dimension up to which the current video block of the one or more blocks of a video region is allowed to be coded using a transform skip mode in which a residual of a prediction error

between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the maximum allowed dimension.

[0007] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes determining that a current video block of visual media data is coded using a transform skip mode; and performing, based on the determining, a conversion between the current video block and a bitstream representation of the visual media data, wherein, during the conversion, the current video block is divided into a plurality of coefficient groups and signaling of a coded block flag for at least one of the plurality of coefficient groups is excluded in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transformation.

[0008] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes determining that a current video block of visual media data is coded using a transform skip mode; and performing, based on the determining, a conversion between the current video block and a bitstream representation of the visual media data, wherein, during the conversion, the current video block is divided into a plurality of coefficient groups, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transformation, and further wherein, during the conversion, a coefficient scanning order of the plurality of coefficient groups is determined based, at least in part, on an indication in the bitstream representation.

[0009] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes using, for encoding a current video block in a video region of a visual media data into a bitstream representation of the visual media data, a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and selecting a context of a sign flag of the current video block according to sign flags of one or more neighboring video blocks in a coefficient group associated with the current video block based upon division of the current video block into a plurality of coefficient groups.

[0010] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes parsing a bitstream representation of a visual media data comprising a video region comprising a current video block to identify a context of a sign flag used in a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and generating the decoded video region from the bitstream representation such that the context of the sign flag is according to sign

flags of one or more neighboring video blocks in a coefficient group associated with the current video block based upon division of the current video block into a plurality of coefficient groups.

[0011] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes determining a position of a current coefficient associated with a current video block of visual media data when the current video block is divided into a plurality of coefficient positions; deriving, at least based on sign flags of one or more neighboring coefficients, a context of a sign flag for the current coefficient; and generating, based on the context, a sign flag for the current coefficient, wherein the sign flag of the current coefficient is used in a transform skip mode in which the current video block is coded without applying a transform.

[0012] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes determining, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, based on satisfying at least one rule that a chroma transform skip mode is applicable on a current video block, wherein, in the chroma transform skip mode a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation; and including a syntax element indicative of the chroma transform skip mode in the bitstream representation.

[0013] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks that at least one rule associated with application of a chroma transform skip mode is satisfied, wherein, in the chroma transform skip mode a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation; and generating a decoded video region from the bitstream representation by decoding the one or more video blocks.

[0014] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes making a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of a transform skip mode to the current video block based on a condition; and including a syntax element indicative of the condition in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation.

[0015] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a condition relating to usage of a transform skip mode in which a residual of a prediction error between

the current video block and a reference video block is represented in the bitstream representation without applying a transform; and generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the condition.

[0016] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes making a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) technique based on an indication of a transform skip mode in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation, wherein, in the QR-BDPCM technique, the residual of the prediction error in a horizontal direction and/or a vertical direction is quantized and entropy-coded; and including, in the bitstream representation, an indication of the selective application of the QR-BDPCM technique.

[0017] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) technique based on an indication of a transform skip mode in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation, wherein, in the QR-BDPCM technique, the residual of the prediction error in a horizontal direction and/or a vertical direction is quantized and entropy-coded; and generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the maximum allowed dimension.

[0018] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes making a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of a separate or dual tree based on a condition; and including a syntax element indicative of the selective application of the separate or the dual tree in the bitstream representation.

[0019] In yet another exemplary aspect, a method for visual media processing is disclosed. The method includes parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a selective application of a separate or dual tree based on or inferred from a condition; and generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the syntax element.

[0020] In yet another example aspect, a video encoder or decoder apparatus comprising a processor configured to implement an above described method is disclosed.

[0021] In another example aspect, a computer readable program medium is disclosed. The medium stores code that embodies processor executable instructions for implementing one of the disclosed methods.

[0022] These, and other, aspects are further described in the present document.

BRIEF DESCRIPTION OF THE DRAWINGS

[0023] FIG. 1 shows an example of intra block copy.

[0024] FIG. 2 shows an example of a block coded in palette mode.

[0025] FIG. 3 shows an example of use of a palette predictor to signal palette entries.

[0026] FIG. 4 shows an example of horizontal and vertical traverse scans.

[0027] FIG. 5 shows an example of coding of palette indices.

[0028] FIG. 6 shows an example of multi-type tree splitting modes.

[0029] FIG. 7 shows an example of samples used to derive parameters in a cross-component linear model (CCLM) prediction mode.

[0030] FIG. 8 shows an exemplary architecture for luma mapping with chroma scaling.

[0031] FIGS. 9A–9E are flowcharts for examples of a video processing methods.

[0032] FIG. 10 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.

[0033] FIG. 11 is a block diagram of an example video processing system in which disclosed techniques may be implemented.

[0034] FIG. 12 is a flowchart of an example method for visual media processing.

[0035] FIG. 13 is a flowchart of an example method for visual media processing.

[0036] FIG. 14 is a flowchart of an example method for visual media processing.

[0037] FIG. 15 is a flowchart of an example method for visual media processing.

[0038] FIG. 16 is a flowchart of an example method for visual media processing.

[0039] FIG. 17 is a flowchart of an example method for visual media processing.

[0040] FIG. 18 is a flowchart of an example method for visual media processing.

[0041] FIG. 19 is a flowchart of an example method for visual media processing.

- [0042] FIG. 20 is a flowchart of an example method for visual media processing.
- [0043] FIG. 21 is a flowchart of an example method for visual media processing.
- [0044] FIG. 22 is a flowchart of an example method for visual media processing.
- [0045] FIG. 23 is a flowchart of an example method for visual media processing.
- [0046] FIG. 24 is a flowchart of an example method for visual media processing.
- [0047] FIG. 25 is a flowchart of an example method for visual media processing.
- [0048] FIG. 26 is a flowchart of an example method for visual media processing.

DETAILED DESCRIPTION

[0049] The present document provides various techniques that can be used by a decoder of image or video bitstreams to improve the quality of decompressed or decoded digital video or images. For brevity, the term “video” is used herein to include both a sequence of pictures (traditionally called video) and individual images. Furthermore, a video encoder may also implement these techniques during the process of encoding in order to reconstruct decoded frames used for further encoding.

[0050] Section headings are used in the present document for ease of understanding and do not limit the embodiments and techniques to the corresponding sections. As such, embodiments from one section can be combined with embodiments from other sections.

1. Summary

[0051] This document is related to video coding/decoding technologies. Specifically, it is related to coefficient coding in a transform skip mode in video coding. It may be applied to the existing video coding standard like HEVC, or the standard (Versatile Video Coding) to be finalized. It may be also applicable to future video coding standards or video codec.

2. Initial Discussion

[0052] 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 [1,2]. 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.

[0053] The latest version of VVC draft, i.e., Versatile Video Coding (Draft 4) could be found at:

[0054] http://phenix.it-sudparis.eu/jvet/doc_end_user/current_document.php?id=5755

[0055] The latest reference software of VVC, named VTM, could be found at:

[0056] https://vcgit.hhi.fraunhofer.de/jvet/VVCSsoftware_VTM/tags/VTM-4.0

Intra block copy

[0057] Intra block copy (IBC), a.k.a. current picture referencing, has been adopted in HEVC Screen Content Coding extensions (HEVC-SCC) and the current VVC test model (VTM-4.0). IBC extends the concept of motion compensation from inter-frame coding to intra-frame coding. As demonstrated in FIG. 1, the current block is predicted by a reference block in the same picture when IBC is applied. The samples in the reference block must have been already reconstructed before the current block is coded or decoded. Although IBC is not so efficient for most camera-captured sequences, it shows significant coding gains for screen content. The reason is that there are lots of repeating patterns, such as icons and text characters in a screen content picture. IBC can remove the redundancy between these repeating patterns effectively. In HEVC-SCC, an inter-coded coding unit (CU) can apply IBC if it chooses the current picture as its reference picture. The MV is renamed as block vector (BV) in this case, and a BV always has an integer-pixel precision. To be compatible with main profile HEVC, the current picture is marked as a “long-term” reference picture in the Decoded Picture Buffer (DPB). It should be noted that similarly, in multiple view/3D video coding standards, the inter-view reference picture is also marked as a “long-term” reference picture.

[0058] Following a BV to find its reference block, the prediction can be generated by copying the reference block. The residual can be got by subtracting the reference pixels from the original signals. Then transform and quantization can be applied as in other coding modes.

[0059] However, when a reference block is outside of the picture, or overlaps with the current block, or outside of the reconstructed area, or outside of the valid area restricted by some constraints, part or all pixel values are not defined. Basically, there are two solutions to handle such a problem. One is to disallow such a situation, e.g. in bitstream conformance. The other is to apply padding for those undefined pixel values. The following sub-sessions describe the solutions in detail.

IBC in HEVC Screen Content Coding extensions

[0060] In the screen content coding extensions of HEVC, when a block uses current picture as reference, it should guarantee that the whole reference block is within the available reconstructed area, as indicated in the following spec text:

[0061] The variables offsetX and offsetY are derived as follows:

$$\text{offsetX} = (\text{ChromaArrayType} == 0) ? 0 : (\text{mvCLX}[0] \& 0x7 ? 2 : 0) \quad (8-106)$$

$$\text{offsetY} = (\text{ChromaArrayType} == 0) ? 0 : (\text{mvCLX}[1] \& 0x7 ? 2 : 0) \quad (8-107)$$

[0062] It is a requirement of bitstream conformance that when the reference picture is the current picture, the luma motion vector mvLX shall obey the following constraints:

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

$$(\text{xPb} + (\text{mvLX}[0] \gg 2) + \text{nPbSw} - 1 + \text{offsetX}) / \text{CtbSizeY} - \text{xCurr} / \text{CtbSizeY} \leq \text{yCurr} / \text{CtbSizeY} - (\text{yPb} + (\text{mvLX}[1] \gg 2) + \text{nPbSh} - 1 + \text{offsetY}) / \text{CtbSizeY} \quad (8-108)$$

[0063] Thus, the case that the reference block overlaps with the current block or the reference block is outside of the picture will not happen. There is no need to pad the reference or prediction block.

IBC in VVC Test Model

[0064] In the current VVC test model, i.e. VTM-4.0 design, the whole reference block should be with the current coding tree unit (CTU) and does not overlap with the current block. Thus, there is no need to pad the reference or prediction block. The IBC flag is coded as a prediction mode of the current CU. Thus, there are totally three prediction modes, MODE_INTRA, MODE_INTER and MODE_IBC for each CU.

IBC Merge mode

[0065] In IBC merge mode, an index pointing to an entry in the IBC merge candidates list is parsed from the bitstream. The construction of the IBC merge list can be summarized according to the following sequence of steps:

[0066] Step 1: Derivation of spatial candidates

[0067] Step 2: Insertion of HMVP candidates

[0068] Step 3: Insertion of pairwise average candidates

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

[0070] After insertion of the spatial candidates, if the IBC merge list size is still smaller than the maximum IBC merge list size, IBC candidates from HMVP table may be inserted. Redundancy check are performed when inserting the HMVP candidates.

[0071] Finally, pairwise average candidates are inserted into the IBC merge list.

[0072] When a reference block identified by a merge candidate is outside of the picture, or overlaps with the current block, or outside of the reconstructed area, or outside of the valid area restricted by some constraints, the merge candidate is called invalid merge candidate.

[0073] It is noted that invalid merge candidates may be inserted into the IBC merge list.

IBC AMVP mode

[0074] In IBC AMVP mode, an AMVP index point to an entry in the IBC AMVP list is parsed from the bitstream. The construction of the IBC AMVP list can be summarized according to the following sequence of steps:

[0075] Step 1: Derivation of spatial candidates

[0076] Check A0, A1 until an available candidate is found.

[0077] Check B0, B1, B2 until an available candidate is found.

[0078] Step 2: Insertion of HMVP candidates

[0079] Step 3: Insertion of zero candidates

[0080] After insertion of the spatial candidates, if the IBC AMVP list size is still smaller than the maximum IBC AMVP list size, IBC candidates from HMVP table may be inserted.

[0081] Finally, zero candidates are inserted into the IBC AMVP list.

Adaptive motion vector resolution (AMVR)

[0082] In HEVC, motion vector differences (MVDs) (between the motion vector and predicted motion vector of a CU) are signalled in units of quarter-luma-sample when use_integer_mv_flag is equal to 0 in the slice header. In VVC, a CU-level adaptive motion vector resolution (AMVR) scheme is introduced. AMVR allows MVD of the CU to be coded in different precision. Dependent on the mode (normal AMVP mode or affine AMVP mode) for the current CU, the MVDs of the current CU can be adaptively selected as follows:

- Normal AMVP mode: quarter-luma-sample, integer-luma-sample or four-luma-sample.
- Affine AMVP mode: quarter-luma-sample, integer-luma-sample or 1/16 luma-sample.

[0083] The CU-level MVD resolution indication is conditionally signalled if the current CU has at least one non-zero MVD component. If all MVD components (that is, both horizontal and vertical MVDs for reference list L0 and reference list L1) are zero, quarter-luma-sample MVD resolution is inferred.

[0084] For a CU that has at least one non-zero MVD component, a first flag is signalled to indicate whether quarter-luma-sample MVD precision is used for the CU. If the first flag is 0, no further signaling is needed and quarter-luma-sample MVD precision is used for the current CU. Otherwise, a second flag is signalled to indicate whether integer-luma-sample or four-luma-sample MVD precision is used for normal AMVP CU. The same second flag is used to indicate whether integer-luma-sample or 1/16 luma-sample MVD precision is used for affine AMVP CU. In order to ensure the reconstructed MV has the intended precision (quarter-luma-sample, integer-luma-sample or four-luma-sample), the motion vector predictors for the CU will be rounded to the same precision as that of the MVD before being added together with the MVD. The motion vector predictors are rounded toward zero (that is, a negative motion vector predictor is rounded toward positive infinity and a positive motion vector predictor is rounded toward negative infinity).

[0085] The encoder determines the motion vector resolution for the current CU using RD check. To avoid always performing CU-level RD check three times for each MVD resolution, in VTM4, the RD check of MVD precisions other than quarter-luma-sample is only invoked conditionally. For normal AMVP mode, the RD cost of quarter-luma-sample MVD precision and integer-luma sample MV precision is computed first. Then, the RD cost of integer-luma-sample MVD precision is compared to that of quarter-luma-sample MVD precision to decide whether it is necessary to further check the RD cost of four-luma-sample MVD precision. When the RD cost for quarter-luma-sample MVD precision

is much smaller than that of the integer-luma-sample MVD precision, the RD check of four-luma-sample MVD precision is skipped. For affine AMVP mode, if affine inter mode is not selected after checking rate-distortion costs of affine merge/skip mode, merge/skip mode, quarter-luma sample MVD precision normal AMVP mode and quarter-luma sample MVD precision affine AMVP mode, then 1/16 luma-sample MV precision and 1-pel MV precision affine inter modes are not checked. Furthermore, affine parameters obtained in quarter-luma-sample MV precision affine inter mode is used as starting search point in 1/16 luma-sample and quarter-luma-sample MV precision affine inter modes.

Palette Mode

[0086] The basic idea behind a palette mode is that the samples in the CU are represented by a small set of representative color values. This set is referred to as the palette. It is also possible to indicate a sample that is outside the palette by signaling an escape symbol followed by (possibly quantized) component values. This is illustrated in FIG. 2.

Palette Mode in HEVC Screen Content Coding extensions (HEVC-SCC)

[0087] In the palette mode in HEVC-SCC, a predictive way is used to code the palette and index map.

Coding of the palette entries

[0088] For coding of the palette entries, a palette predictor is maintained. The maximum size of the palette as well as the palette predictor is signalled in the SPS. In HEVC-SCC, a `palette_predictor_initializer_present_flag` is introduced in the PPS. When this flag is 1, entries for initializing the palette predictor are signalled in the bitstream. The palette predictor is initialized at the beginning of each CTU row, each slice and each tile. Depending on the value of the `palette_predictor_initializer_present_flag`, the palette predictor is reset to 0 or initialized using the palette predictor initializer entries signalled in the PPS. In HEVC-SCC, a palette predictor initializer of size 0 was enabled to allow explicit disabling of the palette predictor initialization at the PPS level.

[0089] For each entry in the palette predictor, a reuse flag is signaled to indicate whether it is part of the current palette. This is illustrated in FIG. 3. The reuse flags are sent using run-length coding of zeros. After this, the number of new palette entries are signaled using exponential Golomb code of order 0. Finally, the component values for the new palette entries are signaled.

Coding of palette indices

[0090] The palette indices are coded using horizontal and vertical traverse scans as shown in FIG. 4. The scan order is explicitly signaled in the bitstream using the `palette_transpose_flag`. For the rest of the subsection it is assumed that the scan is horizontal.

[0091] The palette indices are coded using two main palette sample modes: 'INDEX' and 'COPY_ABOVE'. As explained previously, the escape symbol is also signaled as an 'INDEX' mode and assigned an index equal to the maximum palette size. The mode is signaled using a flag except for the top row or when the previous mode was 'COPY_ABOVE'. In the 'COPY_ABOVE' mode, the palette index of the sample in the row above is copied. In the 'INDEX' mode, the palette index is explicitly signalled. For both 'INDEX' and 'COPY_ABOVE' modes, a run value is signalled which specifies the number of subsequent samples that are also coded using the same mode. When escape symbol is part of the run in 'INDEX' or 'COPY_ABOVE' mode, the escape component values are signalled for each escape symbol. The coding of palette indices is illustrated in FIG. 5.

[0092] This syntax order is accomplished as follows. First the number of index values for the CU is signaled. This is followed by signaling of the actual index values for the entire CU using truncated binary coding. Both the number of indices as well as the index values are coded in bypass mode. This groups the index-related bypass bins together. Then the palette sample mode (if necessary) and run are signaled in an interleaved manner. Finally, the component escape values corresponding to the escape samples for the entire CU are grouped together and coded in bypass mode.

[0093] An additional syntax element, `last_run_type_flag`, is signaled after signaling the index values. This syntax element, in conjunction with the number of indices, eliminates the need to signal the run value corresponding to the last run in the block.

[0094] In HEVC-SCC, the palette mode is also enabled for 4:2:2, 4:2:0, and monochrome chroma formats. The signaling of the palette entries and palette indices is almost identical for all the chroma formats. In case of non-monochrome formats, each palette entry consists of 3 components. For the monochrome format, each palette entry consists of a single component. For subsampled chroma directions, the chroma samples are associated with luma sample indices that are divisible by 2. After reconstructing the palette indices for the CU, if a sample has only a single component associated with it, only the first component of the palette entry is used. The only difference in signaling is for the escape component values. For each escape sample, the number of escape component values signaled may be different depending on the number of components associated with that sample.

Coefficients Coding in Transform Skip mode

[0095] In JVET-M0464 and JVET-N0280, several modifications are proposed on the coefficients coding in transform skip (TS) mode in order to adapt the residual coding to the statistics and signal characteristics of the transform skip levels.

[0096] The proposed modifications are listed as follows.

[0097] **No last significant scanning position:** Since the residual signal reflects the spatial residual after the prediction and no energy compaction by transform is performed for TS, the higher

probability for trailing zeros or insignificant levels at the bottom right corner of the transform block is not given anymore. Thus, last significant scanning position signalling is omitted in this case.

[0098] Subblock CBFs: The absence of the last significant scanning position signalling requires the subblock CBF signalling with coded_sub_block_flag for TS to be modified as follows:

- Due to quantization, the aforementioned sequence of insignificance may still occur locally inside a transform block. Thus, the last significant scanning position is removed as described before and **coded_sub_block_flag is coded for all sub-blocks.**

- The coded_sub_block_flag for the subblock covering the DC frequency position (top-left subblock) presents a special case. In VVC Draft 3, the coded_sub_block_flag for this subblock is never signaled and always inferred to be equal to 1. When the last significant scanning position is located in another subblock, it means that there is at least one significant level outside the DC subblock. Consequently, the DC subblock may contain only zero/non-significant levels although the coded_sub_block_flag for this subblock is inferred to be equal to 1. With the absence of the last scanning position information in TS, the coded_sub_block_flag for each subblock is signaled. This also includes the coded_sub_block_flag for the DC subblock except when all other coded_sub_block_flag syntax elements are already equal to 0. In this case, the DC coded_sub_block_flag is inferred to be equal to 1 (inferDcSbCbf=1). Since there has to be at least one significant level in this DC subblock, the **sig_coeff_flag syntax element for the first position at (0,0) is not signaled and derived to be equal to 1** (inferSbDcSigCoeffFlag=1) instead if all other sig_coeff_flag syntax elements in this DC subblock are equal to 0.

- The context modeling for coded_sub_block_flag is changed. The **context model index is calculated as the sum of the coded_sub_block_flag to the left and the coded_sub_block_flag above** the current subblock instead of and a logical disjunction of both.

[0099] sig_coeff_flag context modelling: The local template in sig_coeff_flag context modeling is modified to only include the neighbor to the left (NB₀) and the neighbor above (NB₁) the current scanning position. The context model offset is just the number of significant neighboring positions sig_coeff_flag[NB₀] + sig_coeff_flag[NB₁]. Hence, the selection of different context sets depending on the diagonal *d* within the current transform block is removed. This results in three context models and a single context model set for coding the sig_coeff_flag flag.

[0100] abs_level_gt1_flag and par_level_flag context modelling: a single context model is employed for abs_level_gt1_flag and par_level_flag.

[0101] abs_remainder coding: Although the empirical distribution of the transform skip residual absolute levels typically still fits a Laplacian or a Geometrical distribution, there exist larger instationarities than for transform coefficient absolute levels. Particularly, the variance within a window

of consecutive realization is higher for the residual absolute levels. This motivates the following modifications of the abs_remainder syntax binarization and context modelling:

- Using a higher cutoff value in the binarization, i.e., the transition point from the coding with sig_coeff_flag, abs_level_gt1_flag, par_level_flag, and abs_level_gt3_flag to the Rice codes for abs_remainder, and dedicated context models for each bin position yields higher compression efficiency. Increasing the cutoff will result in more “greater than X” flags, e.g. introducing **abs_level_gt5_flag, abs_level_gt7_flag, and so on until a cutoff is reached**. The cutoff itself is fixed to 5 (numGtFlags=5).

- The **template for the rice parameter derivation is modified**, i.e., only the neighbor to the left and the neighbor above the current scanning position are considered similar to the local template for sig_coeff_flag context modeling.

[0102] coeff sign flag context modelling: Due to the instationarities inside the sequence of signs and the fact that the prediction residual is often biased, the signs can be coded using context models, even when the global empirical distribution is almost uniformly distributed. A single dedicated context model is used for the coding of the signs and the sign is parsed after sig_coeff_flag to keep all context coded bins together.

Quantized residual Block Differential Pulse-code Modulation(QR-BDPCM)

[0103] In JVET-M0413, a quantized residual block differential pulse-code modulation (QR-BDPCM) is proposed to code screen contents efficiently.

[0104] The prediction directions used in QR-BDPCM can be vertical and horizontal prediction modes. 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. This can be described by the following: 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 signalled:

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

[0105] For horizontal prediction, similar rules apply, and the residual quantized samples are obtained by

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

[0106] The residual quantized samples $\tilde{r}_{i,j}$ are sent to the decoder.

[0107] 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,

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

[0108] For horizontal case,

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

[0109] The inverse quantized residuals, $Q^{-1}(Q(r_{i,j}))$, are added to the intra block prediction values to produce the reconstructed sample values.

[0110] The main benefit of this scheme is that the inverse DPCM can be done on the fly during coefficient parsing simply adding the predictor as the coefficients are parsed or it can be performed after parsing.

[0111] The draft text changes of QR-BDPCM are shown as follows.

7.3.6.5 Coding unit syntax

	Descriptor
<code>coding_unit(x0, y0, cbWidth, cbHeight, treeType) {</code>	
<code>if(tile_group_type != I sps_ibc_enabled_flag) {</code>	
<code>if(treeType != DUAL_TREE_CHROMA)</code>	
<code>cu_skip_flag[x0][y0]</code>	ae(v)
<code>if(cu_skip_flag[x0][y0] == 0 && tile_group_type != I)</code>	
<code>pred_mode_flag</code>	ae(v)
<code>if(((tile_group_type == I && cu_skip_flag[x0][y0] == 0) (tile_group_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)</code>	
<code>pred_mode_ibc_flag</code>	ae(v)
<code>}</code>	
<code>if(CuPredMode[x0][y0] == MODE_INTRA) {</code>	
<code>if(pred_mode_flag == MODE_INTRA && (cIdx == 0) && (cbWidth <= 32) && (CbHeight <= 32)) {</code>	
<code>bdpcm_flag[x0][y0]</code>	ae(v)
<code>if(bdpcm_flag[x0][y0]) {</code>	
<code>bdpcm_dir_flag[x0][y0]</code>	ae(v)
<code>}</code>	

else {	
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 */	
...	
}	

bdpcm_flag[x0][y0] equal to 1 specifies that a bdpcm_dir_flag is present in the coding unit including the luma coding block at the location (x0, y0)

bdpcm_dir_flag[x0][y0] equal to 0 specifies that the prediction direction to be used in a bdpcm block is horizontal, otherwise it is vertical.

Partition Structure

Partitioning of the CTUs using a tree structure

[0112] In HEVC, a CTU is split into CUs by using a quaternary-tree structure denoted as coding tree to adapt to various local characteristics. The decision whether to code a picture area using inter-picture (temporal) or intra-picture (spatial) prediction is made at the leaf CU level. Each leaf CU can be further split into one, two or four PUs according to the PU splitting type. Inside one PU, the same prediction process is applied and the relevant information is transmitted to the decoder on a PU basis. After obtaining the residual block by applying the prediction process based on the PU splitting type, a leaf CU can be partitioned into transform units (TUs) according to another quaternary-tree structure similar to the coding tree for the CU. One of key feature of the HEVC structure is that it has the multiple partition conceptions including CU, PU, and TU.

[0113] In VVC, a quadtree with nested multi-type tree using binary and ternary splits segmentation structure replaces the concepts of multiple partition unit types, i.e. it removes the separation of the CU, PU and TU concepts except as needed for CUs that have a size too large for the maximum transform length, and supports more flexibility for CU partition shapes. In the coding tree structure, a CU can have either a square or rectangular shape. A coding tree unit (CTU) is first partitioned by a quaternary tree (a.k.a. quadtree) structure. Then the quaternary tree leaf nodes can be further partitioned by a multi-type tree structure. As shown in FIG. 6, there are four splitting types in multi-type tree structure, vertical binary splitting (SPLIT_BT_VER), horizontal binary splitting (SPLIT_BT_HOR), vertical ternary splitting (SPLIT_TT_VER), and horizontal ternary splitting (SPLIT_TT_HOR). The multi-type tree leaf nodes are called coding units (CUs), and unless the CU is too large for the maximum transform length, this segmentation is used for prediction and transform processing without any further partitioning. This means that, in most cases, the CU, PU and TU have the same block size in the quadtree with nested multi-type tree coding block structure. The exception occurs when maximum supported transform length is smaller than the width or height of the color component of the CU. In addition, luma and chroma components have separate partition structures on I tiles.

Cross-component linear model prediction

[0114] To reduce the cross-component redundancy, a cross-component linear model (CCLM) prediction mode is used in the VTM4, for which the chroma samples are predicted based on the reconstructed luma samples of the same CU by using a linear model as follows:

$$\text{pred}_C(i,j) = \alpha \cdot \text{rec}_L'(i,j) + \beta$$

[0115] where $\text{pred}_C(i,j)$ represents the predicted chroma samples in a CU and $\text{rec}_L(i,j)$ represents the downsampled reconstructed luma samples of the same CU. Linear model parameter α and β are derived from the relation between luma values and chroma values from two samples, which are

luma sample with minimum sample value and with maximum sample inside the set of downsampled neighboring luma samples, and their corresponding chroma samples. The linear model parameters α and β are obtained according to the following equations.

$$\alpha = \frac{Y_a - Y_b}{X_a - X_b}$$

$$\beta = Y_b - \alpha \cdot X_b$$

[0116] Where Y_a and X_a represent luma value and chroma value of the luma sample with maximum luma sample value. And X_b and Y_b represent luma value and chroma value of the luma sample with minimum luma sample, respectively. FIG. 7 shows an example of the location of the left and above samples and the sample of the current block involved in the CCLM mode.

Luma mapping with chroma scaling (LMCS)

[0117] In VTM4, a coding tool called the luma mapping with chroma scaling (LMCS) is added as a new processing block before the loop filters. LMCS has two main components: 1) in-loop mapping of the luma component based on adaptive piecewise linear models; 2) for the chroma components, luma-dependent chroma residual scaling is applied. FIG. 8 shows the LMCS architecture from decoder's perspective. The shaded blocks in FIG. 8 indicate where the processing is applied in the mapped domain; and these include the inverse quantization, inverse transform, luma intra prediction and adding of the luma prediction together with the luma residual. The unshaded blocks in FIG. 8 indicate where the processing is applied in the original (i.e., non-mapped) domain; and these include loop filters such as deblocking, ALF, and SAO, motion compensated prediction, chroma intra prediction, adding of the chroma prediction together with the chroma residual, and storage of decoded pictures as reference pictures. The light-yellow shaded blocks in FIG. 8 are the new LMCS functional blocks, including forward and inverse mapping of the luma signal and a luma-dependent chroma scaling process. Like most other tools in VVC, LMCS can be enabled/disabled at the sequence level using an SPS flag.

3. Examples of Problems Solved by Embodiments

[0118] Although the coefficient coding in JVET-N0280 can achieve coding benefits on screen content coding, the coefficients coding and transform skip (TS) mode may still have some drawbacks.

- (1) The maximal allowed width or height for TS mode are controlled by one common value in PPS, which may limit the flexibility.
- (2) Every coding group (CG) needs to signal a cbf flag for the TS mode which may increase the overhead cost.
- (3) The coefficient scanning order does not consider the intra prediction mode.
- (4) The sign flag coding only employs one context.
- (5) Transform skip on the chroma component is not supported.

(6) Transform skip flag are applied on all prediction modes which increases the overhead cost and encoding complexity.

4. Examples of Embodiments

[0119] The detailed inventions below should be considered as examples to explain general concepts. These inventions should not be interpreted in a narrow way. Furthermore, these inventions can be combined in any manner.

1. Indications of the maximal allowed width and height for transform skip may be both signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs.
 - a. In one example, the maximal allowed width and height for transform skip may be indicated by different messages signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs.
 - b. In one example, the maximal allowed width and/or height may be first signaled in the SPS/PPS, then updated in the picture header/slice header/tile group header/LCU row/group of LCUs.
2. A TS coded block may be divided into several coefficient groups (CGs) and signaling of the coded block flag (Cbf) flags of at least one CG may be skipped.
 - a. In one example, signaling of all CGs' Cbf flags may be skipped, e.g., for a TS coded blocks.
 - b. In one example, the skipped cbf flags of CGs may be inferred to 1 for the TS mode
 - c. In one example, whether to skip partial or all Cbf flags of CGs may depend on the coded mode.
 - i. In one example, for TS-coded intra blocks, signalling of all Cbf flags of CGs is skipped.
 - d. In one example, the skipped Cbf flag of a CG may be inferred based on
 - i. A message signalled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU
 - ii. The position of the CG
 - iii. Block dimension of current block and/or its neighboring blocks
 - iv. Block shape of current block and/or its neighboring blocks
 - v. The most probable modes of the current block and/or its neighboring blocks
 - vi. Prediction modes (Intra/Inter) of the neighboring blocks of the current block
 - vii. Intra prediction modes of the neighboring blocks of the current block
 - viii. Motion vectors of the neighboring blocks of the current block
 - ix. The indication of QR-BDPCM modes of the neighboring block of the current block

- x. Current quantization parameter of current block and/or that of its neighboring blocks
 - xi. Indication of the color format (such as 4:2:0, 4:4:4)
 - xii. Single/dual coding tree structure
 - xiii. Slice/tile group type and/or picture type
3. The coefficient scanning order in TS coded blocks may be dependent on a message signalled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU.
- a. Alternatively, the CG and/or coefficient scanning order may be dependent on intra prediction mode when TS is employed
 - i. In one example, the scanning order may be vertical if the intra prediction mode is horizontally dominated
 - 1. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 2 to 34.
 - 2. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 2 to 33.
 - ii. In one example, the scanning order may be vertical if the intra prediction mode is vertically dominated
 - 1. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 34-66.
 - 2. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 35-66.
 - iii. In one example, the scanning order may be horizontal if the intra prediction mode is vertically dominated
 - 1. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 34 to 66.
 - 2. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 35 to 66.
 - iv. In one example, the scanning order may be horizontal if the intra prediction mode is horizontally dominated
 - 1. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 2 to 34.
 - 2. In one example, the scanning order may be vertical if the intra prediction mode index ranges from 2 to 33.
4. It is proposed the context of sign flag coding may depend on neighboring information in a coefficient block for TS mode.

- a. In one example, the context of coding the current sign flag may depend on the value of neighboring sign flags for TS mode.
 - i. In one example, the context of coding the current sign flag may depend on the value of sign flags of left and/or above neighbors.
 - 1. In one example, the context of the current sign flag may be derived as $C = (L + A)$, where the C is the context id, L is the sign flag of its left neighbor and A is the sign flag of its above neighbor
 - 2. In one example, the context of the current sign flag may be derived as $C = (L + A * 2)$, where the C is the context id, L is the sign flag of its left neighbor and A is the sign flag of its above neighbor
 - 3. In one example, the context of the current sign flag may be derived as $C = (L * 2 + A)$, where the C is the context id, L is the sign flag of its left neighbor and A is the sign flag of its above neighbor
 - ii. In one example, the context of coding the current sign flag may depend on the value of sign flags of left, above neighbors, and above left neighbor.
 - iii. In one example, the context of coding the current sign flag may depend on the value of sign flags of left, above neighbors, above left neighbor, above right neighbor.
 - b. In one example, the context of coding the current sign flag may depend on the position of the coefficient.
 - i. In one example, the context of sign flag may be different at different positions.
 - ii. In one example, the context of sign flag may depend on $x+y$, where x and y are horizontal and vertical position of a position.
 - iii. In one example, the context of sign flag may depend on $\min(x,y)$, where x and y are horizontal and vertical position of a position.
 - iv. In one example, the context of sign flag may depend on $\max(x,y)$, where x and y are horizontal and vertical position of a position.
5. It is proposed the chroma transform skip mode may be supported.
- a. In one example, usage of the chroma transform skip mode may be based on a message signalled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - b. Alternatively, usage of the chroma transform skip mode may be based on the decoded information of one or multiple representative previously coded blocks in the same color component or other color components.
 - i. In one example, the indication of chroma TS flag may be inferred to false if the indication of the TS flag of a representative block is false. Alternatively, the

- indication of chroma TS flag may be inferred to true if the indication of the TS flag of the representative block is true.
- ii. In one example, the representative block may be a luma block or a chroma block.
 - iii. In one example, the representative block could be any block within the collocated luma block.
 - iv. In one example, the representative block could be one of neighboring chroma blocks of the current chroma block.
 - v. In one example, the representative block may be the block covering the corresponding luma sample of the center chroma sample within current chroma block.
 - vi. In one example, the representative block may be the block covering the corresponding luma sample of the right bottom chroma sample within current chroma block.
6. Whether and/or how to apply the transform skip mode may depends on a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
- a. In one example, the indication of when to and/or how to apply transform skip mode may depend on
 - i. Block dimension of current block and/or its neighboring blocks
 - ii. Block shape of current block and/or its neighboring blocks
 - iii. The most probable modes of the current block and/or its neighboring blocks
 - iv. Prediction modes (Intra/Inter) of the neighboring blocks of the current block
 - v. Intra prediction modes of the neighboring blocks of the current block
 - vi. Motion vectors of the neighboring blocks of the current block
 - vii. The indication of QR-BDPCM modes of the neighboring block of the current block
 - viii. Current quantization parameter of current block and/or that of its neighboring blocks
 - ix. Indication of the color format (such as 4:2:0, 4:4:4)
 - x. Single/dual coding tree structure
 - xi. Slice/tile group type and/or picture type
 - xii. Temporal layer ID
 - b. In one example, transform skip mode may be applied when the prediction mode is an IBC mode and the block width and/or height is smaller/bigger/equal to a threshold
 - i. In one example, the threshold may be 4,8,16 or 32.
 - ii. In one example, the threshold may be signaled in the bitstream.

- iii. In one example, the threshold may be based on
 - 1. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU
 - 2. Block dimension of current block and/or its neighboring blocks
 - 3. Block shape of current block and/or its neighboring blocks
 - 4. The most probable modes of the current block and/or its neighboring blocks
 - 5. Prediction modes (Intra/Inter) of the neighboring blocks of the current block
 - 6. Intra prediction modes of the neighboring blocks of the current block
 - 7. Motion vectors of the neighboring blocks of the current block
 - 8. The indication of QR-BDPCM modes of the neighboring block of the current block
 - 9. Current quantization parameter of current block and/or that of its neighboring blocks
 - 10. Indication of the color format (such as 4:2:0, 4:4:4)
 - 11. Single/dual coding tree structure
 - 12. Slice/tile group type and/or picture type
 - 13. Temporal layer ID
- 7. Whether to signal the indication of TS mode may depend on the decoded/derived intra prediction mode.
 - a. Alternatively, furthermore, it may depend on the allowed intra prediction modes/directions used in QR-BDPCM coded blocks and usage of QR-BDPCM.
 - b. For a decoded or derived intra prediction mode, if it is part of allowed sets of intra prediction modes/directions used in QR-BDPCM coded blocks, the signaling of TS flag may be skipped.
 - i. In one example, if QR-BDPCM is allowed for coding one slice/picture/tile/brick, vertical and horizontal modes are two allowed modes in QR-BDPCM process, and the decoded/derived intra mode is vertical or horizontal mode, then the indication of TS mode is not signaled.
 - c. In one example, transform skip mode may be inferred to be enabled when the indication of QR-BDPCM mode (e.g. `bdpcm_flag`) is 1.
 - d. The above method may be applied based on
 - i. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU
 - ii. Block dimension of current block and/or its neighboring blocks

- iii. Block shape of current block and/or its neighboring blocks
 - iv. The most probable modes of the current block and/or its neighboring blocks
 - v. Prediction modes (Intra/Inter) of the neighboring blocks of the current block
 - vi. Intra prediction modes of the neighboring blocks of the current block
 - vii. Motion vectors of the neighboring blocks of the current block
 - viii. The indication of QR-BDPCM modes of the neighboring block of the current block
 - ix. Current quantization parameter of current block and/or that of its neighboring blocks
 - x. Indication of the color format (such as 4:2:0, 4:4:4)
 - xi. Single/dual coding tree structure
 - xii. Slice/tile group type and/or picture type
 - xiii. Temporal layer ID
8. Whether and/or how to apply QR-BDPCM may depend on the indication of the TS mode.
- a. In one example, the indication of whether to apply QR-BDPCM may be signaled on a transform unit (TU) level instead of being signaled in CU.
 - i. In one example, the indication of whether to apply QR-BDPCM may be signaled after the indication of TS mode is applied to a TU.
 - b. In one example, QR-BDPCM is treated as a special case of TS mode.
 - i. When one block is coded with TS mode, another flag may be further signaled to indicate whether QR-BDPCM or conventional TS mode is applied. If it is coded with QR-BDPCM, the prediction direction used in QR-BDPCM may be further signaled.
 - ii. Alternatively, when one block is coded with TS mode, another flag may be further signaled to indicate which kind of QR-BDPCM (e.g., horizontal/vertical prediction direction-based QR-BDPCM) or conventional TS mode is applied.
 - c. In one example, the indication of whether to QR-BDPCM may be inferred based on the indication of TS mode.
 - i. In one example, the indication of whether to apply QR-BDPCM on a luma and/or chroma block may be inferred to true if the indication of whether to apply the transform skip flag on the same block is true. Alternatively, if the indication of whether to apply transform skip flag on a luma and/or chroma block is true, the indication of whether to apply QR-BDPCM on the same block may be inferred to true.
 - ii. In one example, the indication of whether to apply QR-BDPCM on a luma and/or chroma block may be inferred to false if the indication of whether to apply the

transform skip flag on the same block is false. Alternatively, if the indication of whether to apply the transform skip flag on a luma and/or chroma block is false, the indication of whether to apply QR-BDPCM on the same block may be inferred to false.

9. Whether and/or how to apply a single/dual tree may depend on a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - a. In one example, the indication of whether to apply a single/dual tree may depend whether the current slice/tile/LCU/LCU row/group of LCUs/video data unit is determined as screen contents.
 - i. Furthermore, in one example, whether a slice/tile/LCU/LCU row/group of LCUs/video data unit is determined as screen contents may depend on
 1. a message/flag signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 2. Block dimension of current CTU and/or its neighboring CTUs
 3. Block shape of current CTU and/or its neighboring CTUs
 4. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 5. Indication of the color format (such as 4:2:0, 4:4:4)
 6. Single/dual coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 7. Slice/tile group type and/or picture type
 8. Temporal layer ID
 - b. In one example, the indication of whether to apply a single/dual tree may be inferred, which may depend on
 - i. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - ii. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - iii. Block dimension of current CTU and/or its neighboring CTUs
 - iv. Block shape of current CTU and/or its neighboring CTUs
 - v. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - vi. Indication of the color format (such as 4:2:0, 4:4:4)

- vii. Single/dual coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - viii. Slice/tile group type and/or picture type
 - ix. Temporal layer ID
 - c. In one example, the indication of whether to apply CCLM and/or LMCS may depend on single/dual coding tree structure type
 - i. In one example, the indication CCLM and/or LMCS may be inferred to false when a single/dual tree is used.
 - d. In one example, the indication of whether to apply single/dual tree (i.e., CTU-level adaptation of single tree or dual tree) may depends whether the indication of enabling dual tree (e.g qtbtt_dual_tree_intra_flag) is true.
 - i. Alternatively, furthermore, if the dual tree for a sequence/slice/tile/brick is not enabled (e.g qtbtt_dual_tree_intra_flag is false), the indication of whether to apply single /dual tree in a lower level (e.g., CTU) may be inferred to false
 - 1. Alternatively, if the dual tree is enabled (e.g qtbtt_dual_tree_intra_flag is true), the indication of whether to apply single/dual tree for a video unit smaller than sequence/slice/tile/brick may be signaled.
 - e. In one example, the indication of whether to apply single/dual tree (i.e., CTU-level adaptation of single tree or dual tree) may depends whether a flag at a level higher than CTU level (e.g sps_asdt_flag and/or slice_asdt_flag) is true.
 - i. Alternatively, furthermore, if a flag at a level higher than CTU level (e.g sps_asdt_flag and/or slice_asdt_flag) is false, the indication of whether to apply single/dual tree (i.e., CTU-level adaptation of single tree or dual tree) may be inferred to false.
 - f. The above methods may be also applicable to single tree partition case, or single/dual coding tree structure type.
10. Whether to enable IBC may depend on the coding tree structure type.
- a. In one example, for a given coding tree structure type (e.g., dual tree), the signaling of indication of IBC mode and/or block vectors used in IBC mode may be skipped and inferred.
 - b. In one example, the indication of the IBC mode may be inferred to false when dual coding tree structure type is applied.
 - c. In one example, the indication of the IBC mode of a luma block may be inferred to false when dual coding tree structure type is applied.
 - d. In one example, the indication of the IBC mode of a chroma block may be inferred to false when dual coding tree structure type is applied.

- e. In one example, the indication of the IBC mode may be inferred based on
 - i. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - ii. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - iii. Block dimension of current CTU and/or its neighboring CTUs
 - iv. Block shape of current CTU and/or its neighboring CTUs
 - v. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - vi. Indication of the color format (such as 4:2:0, 4:4:4)
 - vii. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - viii. Slice/tile group type and/or picture type
 - ix. Temporal layer ID
- 11. Whether to enable CCLM may depend on the coding tree structure type.
 - a. In one example, for a given coding tree structure type (e.g., dual tree), the signaling of indication of CCLM mode, and/or other syntax related to CCLM mode, may be skipped and inferred.
 - b. In one example, the indication of the CCLM mode may be inferred to false when dual coding tree structure type is applied.
 - c. In one example, the indication of the CCLM mode, when dual coding tree structure type is applied, may be inferred based on
 - i. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - ii. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - iii. Block dimension of current CTU and/or its neighboring CTUs
 - iv. Block shape of current CTU and/or its neighboring CTUs
 - v. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - vi. Indication of the color format (such as 4:2:0, 4:4:4)
 - vii. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - viii. Slice/tile group type and/or picture type
 - ix. Temporal layer ID
- 12. Whether to enable LMCS for chroma components may depend on the coding tree structure type.

- a. In one example, for a given coding tree structure type (e.g., dual tree), the signaling of indication of LMCS for chroma components, and/or other syntax related to LMCS mode, may be skipped and inferred.
 - b. In one example, the indication of the LMCS for chroma components may be inferred to false when dual coding tree structure type is applied.
 - c. In one example, the indication of the LMCS for chroma components, when dual coding tree structure type is applied, may be inferred based on
 - i. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - ii. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - iii. Block dimension of current CTU and/or its neighboring CTUs
 - iv. Block shape of current CTU and/or its neighboring CTUs
 - v. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - vi. Indication of the color format (such as 4:2:0, 4:4:4)
 - vii. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - viii. Slice/tile group type and/or picture type
 - ix. Temporal layer ID
13. Coding tree structure may depend on whether IBC is used or not.
- a. In one example, the dual tree structure and the IBC method may be not enabled concurrently at a sequence/picture/tile/brick/CTU/VPDU/32x32 block/64x32 block/32x64 block level.
 - b. Alternatively, furthermore, in one example, if IBC method is enabled, the dual tree structure may be disabled at a sequence/picture/tile/brick/CTU/VPDU/32x32 block/64x32 block/32x64 block level.
 - c. In one example, when IBC is used in a region, chroma coding tree structure may be aligned to luma coding tree structure
 - i. In one example, the region may be a sequence/picture/tile/brick/CTU/VPDU/32x32 block/64x32 block/32x64 block.
 - ii. In one example, when a collocated luma block is split into subblocks, the chroma block may be split into subblocks if it is allowed to be split.
 - iii. In one example, whether and how a chroma block is split may be inferred from the coding structure of its collocated luma block.

- iv. In one example, when chroma coding tree structure is inferred from luma coding tree structure, the signals to code the chroma coding tree structure may be skipped.
 - v. In one example, a flag may be used to indicate whether chroma coding structure can be inferred from luma coding structure or not. Signaling of the flag may depend on
 - 1. A message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - 2. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - 3. Block dimension of current CTU and/or its neighboring CTUs
 - 4. Block shape of current CTU and/or its neighboring CTUs
 - 5. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - 6. Indication of the color format (such as 4:2:0, 4:4:4)
 - 7. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - 8. Slice/tile group type and/or picture type
 - 9. Temporal layer ID
14. Whether to enable palette coding mode may depend on the coding tree structure type.
- a. In one example, for a given coding tree structure type (e.g., dual tree), the signaling of indication of palette coding mode may be skipped and inferred.
 - b. In one example, the indication of the palette coding mode may be inferred to false when dual coding tree structure type is applied.
 - c. In one example, the indication of the palette coding mode of a luma block may be inferred to false when dual coding tree structure type is applied.
 - d. In one example, the indication of the palette coding mode of a chroma block may be inferred to false when dual coding tree structure type is applied.
 - e. In one example, the indication of the palette coding mode inferred may be based on
 - i. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - ii. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - iii. Block dimension of current CTU and/or its neighboring CTUs
 - iv. Block shape of current CTU and/or its neighboring CTUs

- v. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - vi. Indication of the color format (such as 4:2:0, 4:4:4)
 - vii. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - viii. Slice/tile group type and/or picture type
 - ix. Temporal layer ID
15. Coding tree structure may depend on whether palette coding mode is used or not.
- a. In one example, when palette coding mode is used in a region, chroma coding tree structure may be aligned to luma coding tree structure
 - i. In one example, the region may be a sequence/picture/tile/brick/CTU/VPDU/32x32 block/64x32 block
 - ii. In one example, when a collocated luma block is split into subblocks, the chroma block may be split into subblocks if it is allowed to be split.
 - iii. In one example, whether and how a chroma block is split may be inferred from the coding structure of its collocated luma block.
 - iv. In one example, when chroma coding tree structure is inferred from luma coding tree structure, the signals to code the chroma coding tree structure may be skipped.
 - v. In one example, a flag may be used to indicate whether chroma coding structure can be inferred from luma coding structure or not. Signaling of the flag may depend on
 - 1. A message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - 2. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - 3. Block dimension of current CTU and/or its neighboring CTUs
 - 4. Block shape of current CTU and/or its neighboring CTUs
 - 5. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - 6. Indication of the color format (such as 4:2:0, 4:4:4)
 - 7. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - 8. Slice/tile group type and/or picture type
 - 9. Temporal layer ID

16. The motion/block vector of a sub-block/sample in a chroma IBC coded block may be derived from the first available IBC-coded sub-region within the collocated luma block.
 - a. In one example, a scanning order of sub-regions within the collocated luma block may be defined, such as raster scanning order.
 - b. In one example, a sub-region may be defined as the minimum coding unit/minimum transform unit.
 - c. In one example, the motion/block vector of whole samples in a chroma IBC mode may be derived based on the motion vector of the most top-left sample with coded in an IBC or inter mode in the collocated luma block.
17. A motion/block vector may be signaled in the chroma IBC mode.
 - a. In one example, the difference between a motion vector and a motion vector predictor may be signaled.
 - i. In one example, the motion vector predictor may be derived based on the motion vectors of a collocated luma block, neighboring luma blocks of the collocated luma block, neighboring chroma blocks of current chroma block.
 1. In one example, the motion/block vector predictor may be derived based on the motion vector of the top-left sample in the collocated luma block.
 2. In one example, the motion/block vector predictor may be derived based on the motion vector of the sample with a center position in the collocated luma block.
 3. In one example, the motion/block vector predictor may be derived based on the motion vector of the most top-left sample with coded in an IBC or inter mode in the collocated luma block.
 - ii. In one example, the motion vector predictor associated one sub-region of luma component may be scaled before being used as a predictor.
 - iii. In one example, the block vector may be derived from motion vectors/blocks vectors of neighboring (adjacent or non-adjacent) chroma blocks.
 - b. In one example, a block vector candidate list may be constructed and an index to the list may be signaled.
 - i. In one example, the candidate list may include motion vectors/block vectors from collocated luma blocks, neighboring luma blocks of the collocated luma blocks, neighboring chroma blocks.
 - c. In one example, the indication of AMVR flag may be inferred
 - i. In one example, the indication of AMVR flag may be inferred to false (0) in a block coded in the chroma IBC mode

- ii. In one example, the indication of motion vector difference may be inferred to integer precision in a block coded in the chroma IBC mode
 - d. In one example, a separate HMVP table may be used on the chroma IBC mode.
 - i. In one example, the size of chroma HMVP table and luma HMVP table may be different.
 - e. In one example, whether to signal a block/motion vector in the chroma IBC mode may be based on
 - i. Whether all sub-regions within the collocated luma block are coded with IBC mode.
 - 1. If yes, no need to signal the chroma block' block vector. Otherwise, chroma block' block vectors may be signaled.
 - ii. Whether all sub-regions within the collocated luma block are coded with IBC mode and all of the associated block vectors are valid.
 - 1. If yes, no need to signal the chroma block' block vector. Otherwise, chroma block' block vectors may be signaled.
 - iii. a message signaled in the SPS/VPS/PPS/picture header/slice header/tile group header/LCU row/group of LCUs/LCU/CU/video data unit.
 - iv. The hash hit ratio of IBC/Inter modes in previous coded pictures/tile/slices/reconstructed region
 - v. Block dimension of current CTU and/or its neighboring CTUs
 - vi. Block shape of current CTU and/or its neighboring CTUs
 - vii. Current quantization parameter of current CTU and/or that of its neighboring CTUs
 - viii. Indication of the color format (such as 4:2:0, 4:4:4)
 - ix. Coding tree structure type of the previous slice/tile/LCU/LCU row/group of LCUs/video data unit
 - x. Slice/tile group type and/or picture type
 - xi. Temporal layer ID
- 18. Single tree and separate/dual tree selection/signaling may be on the level below CTU/CTB
 - a. In one example, single tree and separate/dual tree selection/signaling may be on VPDU level.
 - b. In one example, single tree and separate/dual tree selection/signaling may be on a level where explicit partitioning tree signaling starts.

[0120] The examples described above may be incorporated in the context of the methods described below, e.g., methods 900, 910, 920, 930 and 940, which may be implemented at a video decoder or a video encoder.

[0121] An exemplary method for video processing includes performing a conversion between a current video block and a bitstream representation of a video comprising the current video block, wherein the conversion selectively uses a transform skip mode for the conversion based on an indicator that is included in the bitstream representation, and wherein, using the transform skip mode, a residual of a prediction error of the current video block is represented in the bitstream representation without applying a transformation.

[0122] In some embodiments, the indicator is a maximal allowed width and a maximal allowed height for the transform skip mode.

[0123] In some embodiments, the maximal allowed width and height is signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU) row or a group of LCUs.

[0124] In some embodiments, the maximal allowed width and height is signaled in different messages.

[0125] In some embodiments, the maximal allowed width and height is signaled in a sequence parameter set (SPS) or a picture parameter set (PPS), and wherein an updated value of the maximal allowed width and height is signaled in a picture header, a slice header, a tile group header, a largest coding unit (LCU) row or a group of LCUs.

[0126] FIG. 9A shows a flowchart of another exemplary method for video processing. The method 900 includes, at step 902, determining that a current video block is coded using a transform skip mode.

[0127] The method 900 includes, at step 904, performing, based on the determining, a conversion between the current video block and a bitstream representation of a video comprising the current video block.

[0128] In some embodiments, the current video block is divided into a plurality of coefficient groups, and the bitstream representation omits signaling of a coded block flag for at least one of the plurality of coefficient groups. In an example, the bitstream representation omits the signaling of the coded block flag for each of the plurality of coefficient groups.

[0129] In some embodiments, the coded block flag omitted in the signaling in the bitstream representation is inferred based on one or more of the following: (1) a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs or a coding unit (CU), (2) a position of at least one of the plurality of coefficient groups, (3) a block dimension of the current video block or at least one neighboring block of the current video block, (4) a block shape of the current video block or the at least one neighboring block, (5) a most probable mode of the current video

block or the at least one neighboring block, (6) a prediction mode of the at least one neighboring block, (7) an intra prediction mode of the at least one neighboring block, (8) one or more motion vectors of the at least one neighboring block, (9) an indication of a quantized residual block differential pulse-code modulation (QR-BDPCM) mode of the at least one neighboring block, (10) a current quantization parameter (QP) of the current video block or the at least one neighboring block, (11) an indication of a color format of the current video block, (12) a separate or dual coding tree structure associated with the current video block, or (13) a slice type, a tile group type or a picture type of the current video block.

[0130] In some embodiments, the current video block is divided into a plurality of coefficient groups, and the method 900 further includes the step of determining a coefficient scanning order for the plurality of coefficient groups. In an example, the coefficient scanning order is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs or a coding unit (CU).

[0131] In some embodiments, the plurality of coefficient group or the coefficient scanning order is based on an intra prediction mode of the current video block. In an example, the coefficient scanning order is vertical, and wherein the intra prediction mode is horizontally dominated. In another example, the coefficient scanning order is horizontal, and wherein the intra prediction mode is horizontally dominated. For example, an index of the intra prediction mode ranges from 2 to 33 or from 2 to 34.

[0132] In some embodiments, the plurality of coefficient group or the coefficient scanning order is based on an intra prediction mode of the current video block. In an example, the coefficient scanning order is vertical, and wherein the intra prediction mode is vertically dominated. In another example, the coefficient scanning order is horizontal, and wherein the intra prediction mode is vertically dominated. For example, an index of the intra prediction mode ranges from 34 to 66 or from 35 to 66.

[0133] In some embodiments, a context of a sign flag is based on neighboring information in a coefficient block associated with the current video block. In an example, the context of the sign flag is further based on a position of a coefficient of the coefficient block. In another example, the context of the sign flag is based on $(x+y)$, $\min(x, y)$ or $\max(x, y)$, wherein x and y are a horizontal value and a vertical value of the position of the coefficient, respectively.

[0134] FIG. 9B shows a flowchart of yet another exemplary method for video processing. The method 910 includes, at step 912, determining, for a current video block, that a chroma transform skip mode is applicable.

[0135] The method 910 includes, at step 914, performing, based on the determining, a conversion between the current video block and a bitstream representation of a video comprising the current video block.

[0136] In some embodiments, the determining is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs, a coding unit (CU) or a video data unit.

[0137] In some embodiments, the determining is based on decoded information from one or more representative video blocks that were decoded prior to the performing the conversion, and wherein a sample in each of the one or more representative video blocks and the current video block is based on a common color information. In an example, the one or more representative video blocks comprises a luma block or a chroma block. In another example, the one or more representative video blocks comprises a block within a collocated luma block.

[0138] FIG. 9C shows a flowchart of yet another exemplary method for video processing. The method 920 includes, at step 922, making a decision, during a conversion between a current video block and a bitstream representation of a video comprising the current video block, regarding a selective application of a transform skip mode to the current video block based on a condition.

[0139] The method 920 includes, at step 924, performing, based on the decision, the conversion.

[0140] In some embodiments, the condition is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs, a coding unit (CU) or a video data unit.

[0141] In some embodiments, the condition is based on one or more of the following: (1) a block dimension of the current video block or at least one neighboring block of the current video block, (2) a block shape of the current video block or the at least one neighboring block, (3) a most probable mode of the current video block or the at least one neighboring block, (4) a prediction mode of the at least one neighboring block, (5) an intra prediction mode of the at least one neighboring block, (6) one or more motion vectors of the at least one neighboring block, (7) an indication of a quantized residual block differential pulse-code modulation (QR-BDPCM) mode of the at least one neighboring block, (8) a current quantization parameter (QP) of the current video block or the at least one neighboring block, (9) an indication of a color format of the current video block, (10) a separate or dual coding tree structure associated with the current video block, (11) a slice type, a tile group type or a picture type of the current video block, or (12) a temporal layer identification (ID).

[0142] In some embodiments, the application of the transform skip mode is performed, a prediction mode of the current video block is an inter block copy (IBC) mode, and a width or a height of the current video block is compared to a threshold. In an example, the threshold is signaled in the bitstream representation. In another example, the threshold is 4, 8, 16 or 32.

[0143] In yet another example, the threshold is based on one or more of the following: (1) a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs or a coding unit (CU), (2) a temporal layer identification (ID), (3) a block dimension of the current video block or at least one neighboring block of the current video block, (4) a block shape of the current video block or the at least one neighboring block, (5) a most probable mode of the current video block or the at least one neighboring block, (6) a prediction mode of the at least one neighboring block, (7) an intra prediction mode of the at least one neighboring block, (8) one or more motion vectors of the at least one neighboring block, (9) an indication of a quantized residual block differential pulse-code modulation (QR-BDPCM) mode of the at least one neighboring block, (10) a current quantization parameter (QP) of the current video block or the at least one neighboring block, (11) an indication of a color format of the current video block, (12) a separate or dual coding tree structure associated with the current video block, or (13) a slice type, a tile group type or a picture type of the current video block.

[0144] FIG. 9D shows a flowchart of yet another exemplary method for video processing. The method 930 includes, at step 932, making a decision, during a conversion between a current video block and a bitstream representation of a video comprising the current video block, regarding a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) based on an indication of a transform skip mode in the bitstream representation.

[0145] The method 930 includes, at step 934, performing, based on the decision, the conversion.

[0146] In some embodiments, the indication of the transform skip mode is signaled on a transform unit (TU) level.

[0147] FIG. 9E shows a flowchart of yet another exemplary method for video processing. The method 940 includes, at step 942, making a decision, during a conversion between a current video block and a bitstream representation of a video comprising the current video block, regarding a selective application of a separate or dual tree based on a condition.

[0148] The method 940 includes, at step 944, performing, based on the decision, the conversion.

[0149] In some embodiments, the condition is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs, a coding unit (CU) or a video data unit.

[0150] In some embodiments, the condition is based on determining whether a slice, a tile, a largest coding unit (LCU), an LCU row, a group of LCUs or a video data unit comprising the current video block is screen content. In an example, the determining is based on one or more of the following: (1) a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture

parameter set (PPS), a picture header, a slice header, a tile group header, the LCU, the LCU row, the group of LCUs, the coding unit (CU) or the video data unit, (2) a block dimension of the current video block or at least one neighboring block of the current video block, (3) a block shape of the current video block or the at least one neighboring block, (4) a current quantization parameter (QP) of the current video block or the at least one neighboring block, (5) an indication of a color format of the current video block, (6) a separate or dual coding tree structure associated with the current video block, (7) a slice type, a tile group type or a picture type of the current video block, or (8) a temporal layer identification (ID).

[0151] FIG. 10 is a block diagram of a video processing apparatus 1000. The apparatus 1000 may be used to implement one or more of the methods described herein. The apparatus 1000 may be embodied in a smartphone, tablet, computer, Internet of Things (IoT) receiver, and so on. The apparatus 1000 may include one or more processors 1002, one or more memories 1004 and video processing hardware 1006. The processor(s) 1002 may be configured to implement one or more methods (including, but not limited to, methods 900, 910, 920, 930 and 940) described in the present document. The memory (memories) 1004 may be used for storing data and code used for implementing the methods and techniques described herein. The video processing hardware 1006 may be used to implement, in hardware circuitry, some techniques described in the present document.

[0152] 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. 10.

[0153] In some embodiments, e.g., described in items 5 and 10 et seq above, a method of video processing includes making a determination about whether or not intra block copy mode is applicable for a conversion between a current video block of a video and a bitstream representation based on a type of coding tree structure corresponding to the current video block; and performing the conversion based on the determination.

[0154] In the above method, the bitstream representation excludes an indication of the intra block copy mode. In other words, the bitstream does not carry an explicit signaling of the IBC mode.

[0155] In the above method, the type of coding tree structure is a dual coding tree structure and the determination is that intra block copy mode is not applicable.

[0156] FIG. 11 is a block diagram showing an example video processing system 1100 in which various techniques disclosed herein may be implemented. Various implementations may include some or all of the components of the system 1100. The system 1100 may include input 1102 for receiving video content. The video content may be received in a raw or uncompressed format, e.g., 8 or 10 bit multi-component pixel values, or may be in a compressed or encoded format. The input 1102 may represent a network interface, a peripheral bus interface, or a storage interface. Examples of network

interface include wired interfaces such as Ethernet, passive optical network (PON), etc. and wireless interfaces such as Wi-Fi or cellular interfaces.

[0157] The system 1100 may include a coding component 1104 that may implement the various coding or encoding methods described in the present document. The coding component 1104 may reduce the average bitrate of video from the input 1102 to the output of the coding component 1104 to produce a coded representation of the video. The coding techniques are therefore sometimes called video compression or video transcoding techniques. The output of the coding component 1104 may be either stored, or transmitted via a communication connected, as represented by the component 1106. The stored or communicated bitstream (or coded) representation of the video received at the input 1102 may be used by the component 1108 for generating pixel values or displayable video that is sent to a display interface 1110. The process of generating user-viewable video from the bitstream representation is sometimes called video decompression. Furthermore, while certain video processing operations are referred to as “coding” operations or tools, it will be appreciated that the coding tools or operations are used at an encoder and corresponding decoding tools or operations that reverse the results of the coding will be performed by a decoder.

[0158] Examples of a peripheral bus interface or a display interface may include universal serial bus (USB) or high definition multimedia interface (HDMI) or Displayport, and so on. Examples of storage interfaces include SATA (serial advanced technology attachment), PCI, IDE interface, and the like. The techniques described in the present document may be embodied in various electronic devices such as mobile phones, laptops, smartphones or other devices that are capable of performing digital data processing and/or video display.

[0159] FIG. 12 is a flowchart of an example method for visual media encoding. Steps of this flowchart are discussed in connection with example embodiment 1 discussed in Section 4 of this document. At step 1202, the process determines, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, a maximum allowed dimension up to which a current video block of the one or more video blocks is allowed to be encoded using a transform skip mode such that a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform. At step 1204, the process includes a syntax element indicative of the maximum allowed dimension in the bitstream representation.

[0160] FIG. 13 is a flowchart of an example method for visual media decoding. Steps of this flowchart are discussed in connection with example embodiment 1 discussed in Section 4 of this document. At step 1302, the process parses a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a maximum allowed dimension up to which the current video block of the one or more

blocks of a video region is allowed to be coded using a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform. At step 1304, the process generates a decoded video region from the bitstream representation by decoding the one or more video blocks according to the maximum allowed dimension.

[0161] FIG. 14 is a flowchart of an example method for visual media processing. Steps of this flowchart are discussed in connection with example embodiment 2 discussed in Section 4 of this document. At step 1402, the process determines that a current video block of visual media data is coded using a transform skip mode. At step 1404, the process performs, based on the determining, a conversion between the current video block and a bitstream representation of the visual media data, wherein, during the conversion, the current video block is divided into a plurality of coefficient groups and signaling of a coded block flag for at least one of the plurality of coefficient groups is excluded in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transformation.

[0162] FIG. 15 is a flowchart of an example method for visual media processing. Steps of this flowchart are discussed in connection with example embodiment 3 discussed in Section 4 of this document. At step 1502, the process determines that a current video block of visual media data is coded using a transform skip mode. At step 1504, the process performs, based on the determining, a conversion between the current video block and a bitstream representation of the visual media data, wherein, during the conversion, the current video block is divided into a plurality of coefficient groups, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transformation, and further wherein, during the conversion, a coefficient scanning order of the plurality of coefficient groups is determined based, at least in part, on an indication in the bitstream representation.

[0163] FIG. 16 is a flowchart of an example method for visual media encoding. Steps of this flowchart are discussed in connection with example embodiment 4 discussed in Section 4 of this document. At step 1602, the process uses, for encoding a current video block in a video region of a visual media data into a bitstream representation of the visual media data, a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform. At step 1604, the process selects a context of a sign flag of the current video block according to sign flags of one or more neighboring video blocks in a coefficient group associated with the current video block based upon division of the current video block into a plurality of coefficient groups.

[0164] FIG. 17 is a flowchart of an example method for visual media decoding. Steps of this flowchart are discussed in connection with example embodiment 4 discussed in Section 4 of this document. At step 1702, the process parses a bitstream representation of a visual media data comprising a video region comprising a current video block to identify a context of a sign flag used in a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform. At step 1704, the process generates the decoded video region from the bitstream representation such that the context of the sign flag is according to sign flags of one or more neighboring video blocks in a coefficient group associated with the current video block based upon division of the current video block into a plurality of coefficient groups.

[0165] FIG. 18 is a flowchart of an example method for visual media processing. At step 1802, the process determines a position of a current coefficient associated with a current video block of visual media data when the current video block is divided into a plurality of coefficient positions. At step 1804, the process derives, at least based on sign flags of one or more neighboring coefficients, a context of a sign flag for the current coefficient. At step 1806, the process generates, based on the context, a sign flag for the current coefficient, wherein the sign flag of the current coefficient is used in a transform skip mode in which the current video block is coded without applying a transform.

[0166] FIG. 19 is a flowchart of an example method for visual media encoding. Steps of this flowchart are discussed in connection with example embodiment 5 discussed in Section 4 of this document. At step 1902, the process determines, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, based on satisfying at least one rule that a chroma transform skip mode is applicable on a current video block, wherein, in the chroma transform skip mode a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation. At step 1904, the process includes a syntax element indicative of the chroma transform skip mode in the bitstream representation.

[0167] FIG. 20 is a flowchart of an example method for visual media decoding. Steps of this flowchart are discussed in connection with example embodiment 5 discussed in Section 4 of this document. At step 2002, the process parses a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks that at least one rule associated with application of a chroma transform skip mode is satisfied, wherein, in the chroma transform skip mode a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation. At step 2004, the process generates a decoded video region from the bitstream representation by decoding the one or more video blocks.

[0168] FIG. 21 is a flowchart of an example method for visual media encoding. Steps of this flowchart are discussed in connection with example embodiment 6 discussed in Section 4 of this document. At step 2102, the process makes a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of a transform skip mode to the current video block based on a condition. At step 2104, the process includes a syntax element indicative of the condition in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation.

[0169] FIG. 22 is a flowchart of an example method for visual media decoding. Steps of this flowchart are discussed in connection with example embodiment 6 discussed in Section 4 of this document. At step 2202, the process parses a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a condition relating to usage of a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform. At step 2204, the process generates a decoded video region from the bitstream representation by decoding the one or more video blocks according to the condition.

[0170] FIG. 23 is a flowchart of an example method for visual media encoding. Steps of this flowchart are discussed in connection with example embodiment 8 discussed in Section 4 of this document. At step 2302, the process makes a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) technique based on an indication of a transform skip mode in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation, wherein, in the QR-BDPCM technique, the residual of the prediction error in a horizontal direction and/or a vertical direction is quantized and entropy-coded. At step 2304, the process includes, in the bitstream representation, an indication of the selective application of the QR-BDPCM technique.

[0171] FIG. 24 is a flowchart of an example method for visual media decoding. Steps of this flowchart are discussed in connection with example embodiment 8 discussed in Section 4 of this document. At step 2402, the process parses a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) technique based on an indication of a transform skip mode in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and

a reference video block is represented in the bitstream representation of the visual media data without applying a transformation, wherein, in the QR-BDPCM technique, the residual of the prediction error in a horizontal direction and/or a vertical direction is quantized and entropy-coded. At step 2404, the process generates a decoded video region from the bitstream representation by decoding the one or more video blocks according to the maximum allowed dimension.

[0172] FIG. 25 is a flowchart of an example method for visual media encoding. Steps of this flowchart are discussed in connection with example embodiment 9 discussed in Section 4 of this document. At step 2502, the process makes a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of a separate or dual tree based on a condition. At step 2504, the process includes a syntax element indicative of the selective application of the separate or the dual tree in the bitstream representation.

[0173] FIG. 26 is a flowchart of an example method for visual media decoding. Steps of this flowchart are discussed in connection with example embodiment 9 discussed in Section 4 of this document. At step 2602, the process parses a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a selective application of a separate or dual tree based on or inferred from a condition. At step 2604, the process generates a decoded video region from the bitstream representation by decoding the one or more video blocks according to the syntax element.

[0174] Some embodiments of the present document are now presented in clause-based format.

[0175] A1. A method for visual media encoding, comprising:

[0176] determining, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, a maximum allowed dimension up to which a current video block of the one or more video blocks is allowed to be encoded using a transform skip mode such that a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and

[0177] including a syntax element indicative of the maximum allowed dimension in the bitstream representation.

[0178] A2. A method for visual media decoding, comprising:

[0179] parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a maximum allowed dimension up to which the current video block of the one or more blocks of a video region is allowed to be coded using a transform skip mode in which a residual of a prediction error

between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and

[0180] generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the maximum allowed dimension.

[0181] A3. The method of any one or more of clauses A1-A2, wherein the maximum allowed dimensional includes a maximal allowed width and a maximal allowed height associated with the transform skip mode.

[0182] A4. The method of clause A3, wherein the maximal allowed width and height are signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU) row or a group of LCUs.

[0183] A5. The method of any one or more of clauses A1-A3, wherein the maximal allowed width and the maximal allowed height are signaled in different messages in the bitstream representation.

[0184] A6. The method of any one or more of clauses A1-A2, wherein initial values of the maximal allowed width and the maximal allowed height are signaled in a sequence parameter set (SPS) or a picture parameter set (PPS), and further wherein updated values of the maximal allowed width and the maximal allowed height are signaled in a picture header, a slice header, a tile group header, a largest coding unit (LCU) row or a group of LCUs.

[0185] A7. The method of any one or more of clauses A2-A6, wherein the generating the decoded video region comprises decoding the one or more video blocks without using the transform skip mode.

[0186] A8. The method of any one or more of clauses A2-A6, wherein the generating the decoded video region comprises decoding the one or more video blocks based on usage of the transform skip mode.

[0187] A9. A method for visual media processing, comprising:

[0188] determining that a current video block of visual media data is coded using a transform skip mode; and

[0189] performing, based on the determining, a conversion between the current video block and a bitstream representation of the visual media data, wherein, during the conversion, the current video block is divided into a plurality of coefficient groups and signaling of a coded block flag for at least one of the plurality of coefficient groups is excluded in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transformation.

[0190] A10. The method of clause A9, wherein signaling of a coded block flag for each of the plurality of coefficient groups is excluded in the bitstream representation.

[0191] A11. The method of any one or more of clause A9-A10, wherein the coded block flag for the plurality of coefficient groups excluded in the bitstream representation are inferred as a fixed value.

[0192] A12. The method of clause A11, wherein the fixed value is one.

[0193] A13. The method of any or more of clauses A9-A12, further comprising:

[0194] based on a prediction mode of the current video block, making a determination to selectively enable or disable signaling the coded block flag for the plurality of coefficient groups.

[0195]

[0196] A14. The method of clause A13, wherein, if the prediction mode is an intra prediction mode, signaling of the coded block flag for the plurality of coefficient groups is excluded in the bitstream representation.

[0197] A15. The method of any one or more of clauses A9-A14, wherein the coded block flag excluded in the signaling in the bitstream representation is inferred based on one or more of the following:

[0198] (1) a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs or a coding unit (CU),

[0199] (2) a position of at least one of the plurality of coefficient groups,

[0200] (3) a block dimension of the current video block or at least one neighboring block of the current video block,

[0201] (4) a block shape of the current video block or the at least one neighboring block,

[0202] (5) a most probable mode of the current video block or the at least one neighboring block,

[0203] (6) a prediction mode of the at least one neighboring block,

[0204] (7) an intra prediction mode of the at least one neighboring block,

[0205] (8) one or more motion vectors of the at least one neighboring block,

[0206] (9) an indication of a quantized residual block differential pulse-code modulation (QR-BDPCM) mode of the at least one neighboring block,

[0207] (10) a current quantization parameter (QP) of the current video block or the at least one neighboring block,

[0208] (11) an indication of a color format of the current video block,

[0209] (12) a separate or dual coding tree structure associated with the current video block, or

[0210] (13) a slice type, a tile group type or a picture type of the current video block.

- [0211] A16. A method for visual media processing, further comprising:
- [0212] determining that a current video block of visual media data is coded using a transform skip mode; and
- [0213] performing, based on the determining, a conversion between the current video block and a bitstream representation of the visual media data, wherein, during the conversion, the current video block is divided into a plurality of coefficient groups, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transformation, and further wherein, during the conversion, a coefficient scanning order of the plurality of coefficient groups is determined based, at least in part, on an indication in the bitstream representation.
- [0214] A17. The method of clause A16, wherein the coefficient scanning order is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs or a coding unit (CU).
- [0215] A18. The method of clause A17, wherein, in the transform skip mode, the plurality of coefficient groups or the coefficient scanning order is based on an intra prediction mode of the current video block.
- [0216] A19. The method of clause A18, wherein the coefficient scanning order is vertical, and wherein the intra prediction mode is horizontally dominated.
- [0217] A20. The method of clause A18, wherein the coefficient scanning order is horizontal, and wherein the intra prediction mode is horizontally dominated.
- [0218] A21. The method of any one or more of clauses A19-A20, wherein an index of the intra prediction mode ranges from 2 to 33 or from 2 to 34.
- [0219] A22. The method of clause A18, wherein the coefficient scanning order is vertical, and wherein the intra prediction mode is vertically dominated.
- [0220] A23. The method of clause A18, wherein the coefficient scanning order is horizontal, and wherein the intra prediction mode is vertically dominated.
- [0221] A24. The method of any one or more of clauses A22-A23, wherein an index of the intra prediction mode ranges from 34 to 66 or from 35 to 66.
- [0222] A25. The method of any one or more of clauses A9-A24, wherein the conversion includes generating the bitstream representation from the current video block.

[0223] A65. The method of any one or more of clauses A9-A24, wherein the conversion includes generating pixel values of the current video block from the bitstream representation.

[0224] C1. A method for visual media encoding, comprising:

[0225] determining, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, based on satisfying at least one rule that a chroma transform skip mode is applicable on a current video block, wherein, in the chroma transform skip mode a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation; and

[0226] including a syntax element indicative of the chroma transform skip mode in the bitstream representation.

[0227] C2. A method for visual media decoding, comprising:

[0228] parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks that at least one rule associated with application of a chroma transform skip mode is satisfied, wherein, in the chroma transform skip mode a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation; and

[0229] generating a decoded video region from the bitstream representation by decoding the one or more video blocks.

[0230] C3. The method of any one or more of clauses C1-C2, wherein the determining is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs, a coding unit (CU) or a video data unit associated with a bitstream representation of the visual media data

[0231] C4. The method of any one or more of clauses C2-C3, wherein the determining is based on decoded information from one or more representative video blocks decoded prior to the conversion, and wherein a sample in each of the one or more representative video blocks and the current video block is based on a common color component.

[0232] C5. The method of clause C4, wherein the one or more representative video blocks comprises a luma block or a chroma block.

[0233] C6. The method of clause C4, wherein the one or more representative video blocks comprises a block within a collocated luma block.

[0234] C7. The method of clause C4, wherein the current video block is a current chroma block, and wherein the one or more representative video blocks comprises a neighboring chroma block of the current chroma block.

[0235] C8. The method of clause C4, wherein the current video block is a current chroma block, and wherein the one or more representative video blocks comprises a block covering a corresponding luma sample of a center chroma sample within the current chroma block.

[0236] C9. The method of clause C4, wherein the current video block is a current chroma block, and wherein the one or more representative video blocks comprises a block covering a corresponding luma sample of a right bottom chroma sample within the current chroma block.

[0237] D1. A method for visual media encoding, comprising:

[0238] making a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of a transform skip mode to the current video block based on a condition; and

[0239] including a syntax element indicative of the condition in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation.

[0240] D2. A method for visual media decoding, comprising:

[0241] parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a condition relating to usage of a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and

[0242] generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the condition.

[0243] D3. The method of clause D1, wherein the syntax element is signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs, a coding unit (CU) or a video data unit.

[0244] D4. The method of any one or more of clauses D1-D3, wherein the condition is based on one or more of the following:

[0245] (1) a block dimension of the current video block or at least one neighboring block of the current video block,

- [0246] (2) a block shape of the current video block or the at least one neighboring block,
 - [0247] (3) a most probable mode of the current video block or the at least one neighboring block,
 - [0248] (4) a prediction mode of the at least one neighboring block,
 - [0249] (5) an intra prediction mode of the at least one neighboring block,
 - [0250] (6) one or more motion vectors of the at least one neighboring block,
 - [0251] (7) an indication of a quantized residual block differential pulse-code modulation (QR-BDPCM) mode of the at least one neighboring block,
 - [0252] (8) a current quantization parameter (QP) of the current video block or the at least one neighboring block,
 - [0253] (9) an indication of a color format of the current video block,
 - [0254] (10) a separate or dual coding tree structure associated with the current video block,
 - [0255] (11) a slice type, a tile group type or a picture type of the current video block, or
 - [0256] (12) a temporal layer identification (ID).
- [0257] D5. The method of clause D4, wherein the transform skip mode is applied when a prediction mode of the current video block is an inter block copy (IBC) mode, and further wherein a width or a height of the current video block satisfies a threshold.
- [0258] D6. The method of clause D5, wherein the threshold is signaled in the bitstream representation.
- [0259] D7. The method of clause D6, wherein a value of the threshold is 4, 8, 16 or 32.
- [0260] D8. The method of any one or more of clauses D5-D7, wherein the threshold is based on one or more of the following:
- [0261] (1) a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs or a coding unit (CU),
 - [0262] (2) a temporal layer identification (ID),
 - [0263] (3) a block dimension of the current video block or at least one neighboring block of the current video block,
 - [0264] (4) a block shape of the current video block or the at least one neighboring block,
 - [0265] (5) a most probable mode of the current video block or the at least one neighboring block,
 - [0266] (6) a prediction mode of the at least one neighboring block,

- [0267] (7) an intra prediction mode of the at least one neighboring block,
- [0268] (8) one or more motion vectors of the at least one neighboring block,
- [0269] (9) an indication of a quantized residual block differential pulse-code modulation (QR-BDPCM) mode of the at least one neighboring block,
- [0270] (10) a current quantization parameter (QP) of the current video block or the at least one neighboring block,
- [0271] (11) an indication of a color format of the current video block,
- [0272] (12) a separate or dual coding tree structure associated with the current video block, or
- [0273] (13) a slice type, a tile group type or a picture type of the current video block.
- [0274] E1. A method for visual media encoding, comprising:
- [0275] making a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) technique based on an indication of a transform skip mode in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation, wherein, in the QR-BDPCM technique, the residual of the prediction error in a horizontal direction and/or a vertical direction is quantized and entropy-coded; and
- [0276] including, in the bitstream representation, an indication of the selective application of the QR-BDPCM technique.
- [0277] E2. A method for visual media decoding, comprising:
- [0278] parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a selective application of quantized residual block differential pulse-code modulation (QR-BDPCM) technique based on an indication of a transform skip mode in the bitstream representation, wherein, in the transform skip mode, a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation of the visual media data without applying a transformation, wherein, in the QR-BDPCM technique, the residual of the prediction error in a horizontal direction and/or a vertical direction is quantized and entropy-coded; and
- [0279] generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the maximum allowed dimension.

[0280] E3. The method of any one or more of clauses E1-E2, wherein the indication of the transform skip mode is signaled at a transform unit (TU) level associated with the current video block.

[0281] E4. The method of any one or more of clauses E1-E3, wherein, in the bitstream representation, the syntax element indicative of application of the QR-BDPCM technique on the current video block is included after the indication of the transform skip mode.

[0282] E5. The method of clause E4, wherein, a first flag in the bitstream representation is used to indicate the transform skip mode and a second flag in the bitstream representation is used to indicate the selective application of the QR-BDPCM technique.

[0283] E6. The method of clause E5, wherein a third flag is used to indicate application of a horizontal prediction direction-based type QR-BDPCM technique and a fourth flag is used to indicate application of a vertical prediction direction-based type QR-BDPCM technique.

[0284] E7. The method of clause E5, wherein a value of the second flag is inferred from a value of the first flag.

[0285] E8. The method of clause E5, wherein a value of the first flag is not signaled and inferred from a value of the second flag.

[0286] E9. The method of clause E8, wherein the value of the first flag is not signaled and inferred as true, when the value of the second flag is true.

[0287] E10. The method of clause E7, wherein the first flag and the second flag take on Boolean values, further comprising:

[0288] upon determining that the value of the first flag is a Boolean true, inferring the value of the second flag as a Boolean true.

[0289] E11. The method of clause E7, wherein the first flag and the second flag take on Boolean values, further comprising:

[0290] upon determining that the value of the first flag is a Boolean false, inferring the value of the second flag as a Boolean false.

[0291] F1. A method for visual media encoding, comprising:

[0292] making a decision, for encoding one or more video blocks in a video region of a visual media data into a bitstream representation of the visual media data, regarding a selective application of a separate or dual tree based on a condition; and

[0293] including a syntax element indicative of the selective application of the separate or the dual tree in the bitstream representation.

[0294] F2. A method for visual media decoding, comprising:

[0295] parsing a syntax element from a bitstream representation of a visual media data comprising a video region comprising one or more video blocks, wherein the syntax element is indicative of a selective application of a separate or dual tree based on or inferred from a condition; and

[0296] generating a decoded video region from the bitstream representation by decoding the one or more video blocks according to the syntax element.

[0297] F3. The method of any one or more of clauses F1-F2, wherein the condition is based on a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, a largest coding unit (LCU), an LCU row, a group of LCUs, a coding unit (CU) or a video data unit.

[0298] F4. The method of any one or more of clauses F1-F2, wherein the condition is based on determining whether a slice, a tile, a largest coding unit (LCU), an LCU row, a group of LCUs or a video data unit comprising the current video block is screen content.

[0299] F5. The method of clause F3, wherein the condition includes one or more of the following:

[0300] (1) a message signaled in a sequence parameter set (SPS), a video parameter set (VPS), a picture parameter set (PPS), a picture header, a slice header, a tile group header, the LCU, the LCU row, the group of LCUs, the coding unit (CU) or the video data unit,

[0301] (2) a block dimension of the current video block or at least one neighboring block of the current video block,

[0302] (3) a block shape of the current video block or the at least one neighboring block,

[0303] (4) a current quantization parameter (QP) of the current video block or the at least one neighboring block,

[0304] (5) an indication of a color format of the current video block,

[0305] (6) a separate or dual coding tree structure associated with the current video block,

[0306] (7) a slice type, a tile group type or a picture type of the current video block, or

[0307] (8) a temporal layer identification (ID).

[0308] F6. A video encoder apparatus comprising a processor configured to implement a method recited in any one or more of clauses A1-F5.

[0309] F7. A video decoder apparatus comprising a processor configured to implement a method recited in any one or more of clauses A1-F5.

[0310] F8. A computer readable medium having code stored thereon, the code embodying processor-executable instructions for implementing a method recited in any of or more of clauses A1-F5.

- [0311] B1. A method for visual media encoding, comprising:
- [0312] using, for encoding a current video block in a video region of a visual media data into a bitstream representation of the visual media data, a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and
- [0313] selecting a context of a sign flag of the current video block according to sign flags of one or more neighboring video blocks in a coefficient group associated with the current video block based upon division of the current video block into a plurality of coefficient groups.
- [0314] B2. A method for visual media decoding, comprising:
- [0315] parsing a bitstream representation of a visual media data comprising a video region comprising a current video block to identify a context of a sign flag used in a transform skip mode in which a residual of a prediction error between the current video block and a reference video block is represented in the bitstream representation without applying a transform; and
- [0316] generating the decoded video region from the bitstream representation such that the context of the sign flag is according to sign flags of one or more neighboring video blocks in a coefficient group associated with the current video block based upon division of the current video block into a plurality of coefficient groups.
- [0317] B3. The method of any one or more of clauses B1-B2, wherein the one or more neighboring video blocks of the current video block include: left neighbors and/or above neighbors and/or above left neighbors and/or above right neighbors.
- [0318] B4. The method of any one or more of clauses B1-B3, wherein a relationship between the context of the sign flag of the current video block and a sign flag of the one or more neighboring video blocks is expressed as:
- [0319] $C = (L + A)$, where C is a context id of the context of the sign flag of the current video block, L is a sign flag of a left neighbor of the current video block, and A is a sign flag of an above neighbor of the current video block.
- [0320] B5. The method of any one or more of clauses B1-B3, wherein a relationship between the context of the sign flag of the current video block and a sign flag of the one or more neighboring video blocks is expressed as:
- [0321] $C = (L + A * 2)$, where C is a context id of the context of the sign flag of the current video block, L is a sign flag of a left neighbor of the current video block, and A is a sign flag of an above neighbor of the current video block.

[0322] B6. The method of any one or more of clauses B1-B3, wherein a relationship between the context of the sign flag of the current video block and a sign flag of the one or more neighboring video blocks is expressed as:

[0323] $C = (L * 2 + A)$, where C is a context id of the context of the sign flag of the current video block, L is a sign flag of a left neighbor of the current video block, and A is a sign flag of an above neighbor of the current video block.

[0324] B7. The method of any one or more of clauses B1-B3, wherein the context of the sign flag of the current video block is based on the sign flags of the one or more neighboring video blocks in accordance with a following rule:

[0325] if sign flags of a pair of neighboring video blocks are both negative, the rule specifies selecting a first context of the sign flag.

[0326] B8. A method for visual media processing, comprising:

[0327] determining a position of a current coefficient associated with a current video block of visual media data when the current video block is divided into a plurality of coefficient positions;

[0328] deriving, at least based on sign flags of one or more neighboring coefficients, a context of a sign flag for the current coefficient; and

[0329] generating, based on the context, a sign flag for the current coefficient, wherein the sign flag of the current coefficient is used in a transform skip mode in which the current video block is coded without applying a transform.

[0330] B9. The method of clause B8, wherein the one or more neighboring coefficients of the current video block include: left neighbors and/or above neighbors.

[0331] B10. The method of clause B8, wherein the context of the sign flag for the current coefficient is further based on the position of the current coefficient.

[0332] B11. The method of clause B8, wherein the visual media processing comprises decoding the current video block from the bitstream representation.

[0333] B12. The method of clause B8, wherein the visual media processing comprises encoding the current video block into the bitstream representation.

[0334] B13. The method of any one or more of clauses B1-B3 or B8-B12, wherein the context of the sign flag of the current video block is based on the sign flags of the one or more neighboring video blocks in accordance with a following rule:

[0335] if only one sign flag of a pair of neighboring video blocks is negative, the rule specifies selecting a second context of the sign flag.

[0336] B14. The method of any one or more of clauses B1-B3 or B8-B12, wherein the context of the sign flag of the current video block is based on the sign flags of the one or more neighboring video blocks in accordance with a following rule:

[0337] if sign flags of a pair of neighboring video blocks are both positive, the rule specifies selecting a third context of the sign flag.

[0338] B15. The method of any one or more of clauses B1-B3 or B8-B12, wherein the context of the sign flag of the current video block is further based on a position of a coefficient group in the plurality of the coefficient groups.

[0339] B16. The method of clause B10, wherein the context of the sign flag is based on at least one of: $(x+y)$, $\min(x, y)$ or $\max(x, y)$, wherein x and y are a horizontal value and a vertical value of the position of the coefficient group, respectively.

[0340] B17. A video encoder apparatus comprising a processor configured to implement a method recited in any one or more of clauses B1-B16.

[0341] B18. A video decoder apparatus comprising a processor configured to implement a method recited in any one or more of clauses B1-B16.

[0342] B19. A computer readable medium having code stored thereon, the code embodying processor-executable instructions for implementing a method recited in any of or more of clauses B1-B19.

[0343] In the present document, the term “video processing” or “visual media processing” may refer to video encoding, video decoding, video compression or video decompression. For example, video compression algorithms may be applied during conversion from pixel representation of a video to a corresponding bitstream representation or vice versa. The bitstream representation of a current video block may, for example, correspond to bits that are either co-located or spread in different places within the bitstream, as is defined by the syntax. For example, a macroblock may be encoded in terms of transformed and coded error residual values and also using bits in headers and other fields in the bitstream.

[0344] 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.

[0345] 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.

[0346] 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.

[0347] 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).

[0348] 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.

[0349] It is intended that the specification, together with the drawings, be considered exemplary only, where exemplary means an example. As used herein, the singular forms “a”, “an” and “the” are intended to include the plural forms as well, unless the context clearly indicates otherwise. Additionally, the use of “or” is intended to include “and/or”, unless the context clearly indicates otherwise.

[0350] 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.

[0351] 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.

[0352] 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.

5. Additional embodiments of the disclosed technology

5.1 Dual tree related embodiment

[0353] The changes on top of the draft provided by JVET- N1001-v7 are highlighted in bolded, underlined, italicized text. The capitalized, bolded font appearing below and elsewhere in this document indicates text that will be potentially deleted from the VVC standard.

5.1.1 Embodiment #1

seq_parameter_set_rbsp() {	Descriptor
.....	
qtbtt_dual_tree_intra_flag	u(1)

if(qtbtt_dual_tree_intra_flag)	
sps_asdt_flag	u(1)
partition_constraints_override_enabled_flag	u(1)
.....	

5.1.2 Embodiment #2

7.3.2.3 Sequence parameter set RBSP syntax

seq_parameter_set_rbsp() {	Descriptor
sps_decoding_parameter_set_id	u(4)
...	
sps_mmvd_enabled_flag	u(1)
...	
if(sps_mmvd_enabled_flag)	
sps_fpel_mmvd_enabled_flag	u(1)
<u>sps_asdt_flag</u>	<u>u(1)</u>
sps_triangle_enabled_flag	u(1)
...	
}	

5.1.3 Embodiment #3

7.3.5 Slice header syntax

7.3.5.1 General slice header syntax

slice_header() {	Descriptor
slice_pic_parameter_set_id	ue(v)
...	
if (slice_type != I) {	
...	
}	
if((weighted_pred_flag && slice_type == P) (weighted_bipred_flag && slice_type == B))	
...	
} else if (sps_ibc_enabled_flag)	
six_minus_max_num_merge_cand	ue(v)
<u>if(sps_asdt_flag)</u>	
<u>slice_asdt_flag</u>	<u>u(1)</u>
slice_qp_delta	se(v)
...	

}	
---	--

sps_asdt_flag equal to 1 specifies that the proposed method can be enabled on the current sequence. It equals to 0 specifies that the proposed method cannot be enabled on the current sequence. When **sps_ASDT_flag** is not present, it is inferred to be equal to 0.

slice_asdt_flag equal to 1 specifies that the proposed method can be enabled on the current slice. It equals to 0 specifies that the proposed method cannot be enabled on the current slice. When **slice_ASDT_flag** is not present, it is inferred to be equal to 0.

7.3.7.2 Coding tree unit syntax

coding_tree_unit() {	Descriptor
xCtb = (CtbAddrInRs % PicWidthInCtbsY) << CtbLog2SizeY	
yCtb = (CtbAddrInRs / PicWidthInCtbsY) << CtbLog2SizeY	
...	
if(slice_asdt_flag)	
ctu_dual_tree_intra_flag	ae(v)
if(slice_type == I && QTBT_DUAL_TREE_INTRA_FLAG && ctu_dual_tree_intra_flag)	
dual_tree_implicit_qt_split (xCtb, yCtb, CtbSizeY, 0)	
Else	
coding_tree(xCtb, yCtb, CtbSizeY, CtbSizeY, 1, 0, 0, 0, 0, 0, SINGLE_TREE)	
}	

ctu_dual_tree_intra_flag equal to 1 specifies that the current CTU employs dual tree coding structure. It equals to 0 specifies that the current CTU employs single tree coding structure. When **ctu_dual_tree_intra_flag** is not present, it is inferred to be equal to 0.

5.1.4 Embodiment #4

7.3.5 Slice header syntax

7.3.5.1 General slice header syntax

slice_header() {	Descriptor
slice_pic_parameter_set_id	ue(v)
...	
if (slice_type != I) {	
...	
}	
if((weighted_pred_flag && slice_type == P) (weighted_bipred_flag && slice_type == B))	
...	
} else if (sps_ibc_enabled_flag)	

six_minus_max_num_merge_cand	ue(v)
if(sps_asdt_flag && qtbtt_dual_tree_intra_flag)	
slice_asdt_flag	u(1)
slice_qp_delta	se(v)
...	
}	

sps_asdt_flag equal to 1 specifies that the proposed method can be enabled on the current sequence. It equals to 0 specifies that the proposed method cannot be enabled on the current sequence. When **sps_ASDT_flag** is not present, it is inferred to be equal to 0.

slice_asdt_flag equal to 1 specifies that the proposed method can be enabled on the current slice. It equals to 0 specifies that the proposed method cannot be enabled on the current slice. When **slice_ASDT_flag** is not present, it is inferred to be equal to 0.

7.3.7.2 Coding tree unit syntax

coding_tree_unit() {	Descriptor
xCtb = (CtbAddrInRs % PicWidthInCtbsY) << CtbLog2SizeY	
yCtb = (CtbAddrInRs / PicWidthInCtbsY) << CtbLog2SizeY	
...	
if(slice_asdt_flag)	
ctu_dual_tree_intra_flag	ue(v)
if(slice_type == I && qtbtt_dual_tree_intra_flag && ctu_dual_tree_intra_flag)	
dual_tree_implicit_qt_split (xCtb, yCtb, CtbSizeY, 0)	
else	
coding_tree(xCtb, yCtb, CtbSizeY, CtbSizeY, 1, 0, 0, 0, 0, 0, SINGLE_TREE)	
}	

ctu_dual_tree_intra_flag equal to 1 specifies that the current CTU employs dual tree coding structure. It equals to 0 specifies that the current CTU employs single tree coding structure. When **ctu_dual_tree_intra_flag** is not present, it is inferred to be equal to 0.

5.2 IBC related embodiment

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 && !(cbWidth == 4 && cbHeight == 4 && !sps_ibc_enabled_flag))	
cu_skip_flag[x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] == 0 && slice_type != I && !(cbWidth == 4 && cbHeight == 4))	

pred_mode_flag	ac(v)
<pre> if(((slice_type == I && cu_skip_flag[x0][y0] == 0) (slice_type != I && (CuPredMode[x0][y0] != MODE_INTRA (cbWidth == 4 && cbHeight == 4 && cu_skip_flag[x0][y0] == 0)))) && sps_ibc_enabled_flag && (cbWidth != 128 cbHeight != 128) && (treeType != DUAL_TREE_CHROMA) </pre>	
pred_mode_ibc_flag	ac(v)
}	
...	
}	

8.6 Decoding process for coding units coded in IBC prediction mode

8.6.1 General decoding process for coding units coded in IBC prediction mode

[0354] Inputs to this process are:

- a luma location (xCb, yCb) specifying the top-left sample of the current coding block relative to the top-left luma sample of the current picture,
- a variable cbWidth specifying the width of the current coding block in luma samples,
- a variable cbHeight specifying the height of the current coding block in luma samples,
- a variable treeType specifying whether a single or a dual tree is used and if a dual tree is used, it specifies whether the current tree corresponds to the luma or chroma components.

[0355] Output of this process is a modified reconstructed picture before in-loop filtering.

[0356] The derivation process for quantization parameters as specified in clause 8.7.1 is invoked with the luma location (xCb, yCb), the width of the current coding block in luma samples cbWidth and the height of the current coding block in luma samples cbHeight, and the variable treeType as inputs.

[0357] The decoding process for coding units coded in ibc prediction mode consists of the following ordered steps:

1. The motion vector components of the current coding unit are derived as follows:

1. IF TREETYPE IS EQUAL TO SINGLE_TREE OR DUAL_TREE_LUMA, THE FOLLOWING APPLIES:

- The derivation process for motion vector components as specified in clause 8.6.2.1 is invoked with the luma coding block location (xCb, yCb), the luma coding block width cbWidth and the luma coding block height cbHeight as inputs, and the luma motion vector mvL[0][0] as output.
- When treeType is equal to SINGLE_TREE, the derivation process for chroma motion vectors in clause 8.6.2.5 is invoked with luma motion vector mvL[0][0] as input, and chroma motion vector mvC[0][0] as output.

- The number of luma coding subblocks in horizontal direction numSbX and in vertical direction numSbY are both set equal to 1.

1. OTHERWISE, IF TREETYPE IS EQUAL TO DUAL_TREE_CHROMA, THE FOLLOWING APPLIES:

- THE NUMBER OF LUMA CODING SUBBLOCKS IN HORIZONTAL DIRECTION NUMSBX AND IN VERTICAL DIRECTION NUMSBY ARE DERIVED AS FOLLOWS:

$$\text{NUMSBX} = (\text{CBWIDTH} \gg 2) \quad (8-871)$$

$$\text{NUMSBY} = (\text{CBHEIGHT} \gg 2) \quad (8-872)$$

- THE CHROMA MOTION VECTORS MVC[XSBIDX][YSBIDX] ARE DERIVED AS FOLLOWS FOR XSBIDX = 0..NUMSBX – 1, YSBIDX = 0..NUMSBY – 1:

- THE LUMA MOTION VECTOR MVL[XSBIDX][YSBIDX] IS DERIVED AS FOLLOWS:

- THE LOCATION (XCUY, YCUY) OF THE COLLOCATED LUMA CODING UNIT IS DERIVED AS FOLLOWS:

$$\text{XCUY} = \text{XCB} + \text{XSBIDX} * 4 \quad (8-873)$$

$$\text{YCUY} = \text{YCB} + \text{YSBIDX} * 4 \quad (8-874)$$

- IF CUPREDMODE[XCUY][YCUY] IS EQUAL TO MODE_INTRA, THE FOLLOWING APPLIES.

$$\text{MVL}[\text{XSBIDX}][\text{YSBIDX}][0] = 0 \quad (8-875)$$

$$\text{MVL}[\text{XSBIDX}][\text{YSBIDX}][1] = 0 \quad (8-876)$$

$$\text{PREDFLAGL0}[\text{XSBIDX}][\text{YSBIDX}] = 0 \quad (8-877)$$

$$\text{PREDFLAGL1}[\text{XSBIDX}][\text{YSBIDX}] = 0 \quad (8-878)$$

- OTHERWISE (CUPREDMODE[XCUY][YCUY] IS EQUAL TO MODE_IBC), THE FOLLOWING APPLIES:

$$\text{MVL}[\text{XSBIDX}][\text{YSBIDX}][0] = \text{MVL0}[\text{XCUY}][\text{YCUY}][0] \quad (8-879)$$

$$\text{MVL}[\text{XSBIDX}][\text{YSBIDX}][1] = \text{MVL0}[\text{XCUY}][\text{YCUY}][1] \quad (8-880)$$

$$\text{PREDFLAGL0}[\text{XSBIDX}][\text{YSBIDX}] = 1 \quad (8-881)$$

$$\text{PREDFLAGL1}[\text{XSBIDX}][\text{YSBIDX}] = 0 \quad (8-882)$$

- THE DERIVATION PROCESS FOR CHROMA MOTION VECTORS IN CLAUSE 8.6.2.5 IS INVOKED WITH MVL[XSBIDX][YSBIDX] AS INPUTS, AND MVC[XSBIDX][YSBIDX] AS OUTPUT.

- IT IS A REQUIREMENT OF BITSTREAM CONFORMANCE THAT THE CHROMA MOTION VECTOR $MVC[XSBIDX][YSBIDX]$ SHALL OBEY THE FOLLOWING CONSTRAINTS:
 - WHEN THE DERIVATION PROCESS FOR BLOCK AVAILABILITY AS SPECIFIED IN CLAUSE 6.4.X [ED. (BB): NEIGHBOURING BLOCKS AVAILABILITY CHECKING PROCESS TBD] IS INVOKED WITH THE CURRENT CHROMA LOCATION (XCURR, YCURR) SET EQUAL TO (XCB / SUBWIDTHC, YCB / SUBHEIGHTC) AND THE NEIGHBOURING CHROMA LOCATION (XCB / SUBWIDTHC + (MVC[XSBIDX][YSBIDX][0] >> 5), YCB / SUBHEIGHTC + (MVC[XSBIDX][YSBIDX][1] >> 5)) AS INPUTS, THE OUTPUT SHALL BE EQUAL TO TRUE.
 - WHEN THE DERIVATION PROCESS FOR BLOCK AVAILABILITY AS SPECIFIED IN CLAUSE 6.4.X [ED. (BB): NEIGHBOURING BLOCKS AVAILABILITY CHECKING PROCESS TBD] IS INVOKED WITH THE CURRENT CHROMA LOCATION (XCURR, YCURR) SET EQUAL TO (XCB / SUBWIDTHC, YCB / SUBHEIGHTC) AND THE NEIGHBOURING CHROMA LOCATION (XCB / SUBWIDTHC + (MVC[XSBIDX][YSBIDX][0] >> 5) + CBWIDTH / SUBWIDTHC – 1, YCB / SUBHEIGHTC + (MVC[XSBIDX][YSBIDX][1] >> 5) + CBHEIGHT / SUBHEIGHTC – 1) AS INPUTS, THE OUTPUT SHALL BE EQUAL TO TRUE.
 - ONE OR BOTH OF THE FOLLOWING CONDITIONS SHALL BE TRUE:
 - (MVC[XSBIDX][YSBIDX][0] >> 5) + XSBIDX * 2 + 2 IS LESS THAN OR EQUAL TO 0.
 - (MVC[XSBIDX][YSBIDX][1] >> 5) + YSBIDX * 2 + 2 IS LESS THAN OR EQUAL TO 0.
- 2. The prediction samples of the current coding unit are derived as follows:
 - IF TREETYPE IS EQUAL TO SINGLE_TREE OR DUAL_TREE_LUMA, THE PREDICTION SAMPLES OF THE CURRENT CODING UNIT ARE DERIVED AS FOLLOWS:
 - The decoding process for ibc blocks as specified in clause 8.6.3.1 is invoked with the luma coding block location (xCb, yCb), the luma coding block width cbWidth and the luma coding block height cbHeight, the number of luma coding subblocks in horizontal direction numSbX and in vertical direction numSbY, the luma motion

- vectors $mvL[xSbIdx][ySbIdx]$ with $xSbIdx = 0..numSbX - 1$, and $ySbIdx = 0..numSbY - 1$, the variable $cIdx$ set equal to 0 as inputs, and the ibc prediction samples ($predSamples$) that are an $(cbWidth) \times (cbHeight)$ array $predSamples_L$ of prediction luma samples as outputs.
- Otherwise if $treeType$ is equal to **SINGLE_TREE OR DUAL_TREE_CHROMA**, the prediction samples of the current coding unit are derived as follows:
 - The decoding process ibc blocks as specified in clause 8.6.3.1 is invoked with the luma coding block location (xCb, yCb) , the luma coding block width $cbWidth$ and the luma coding block height $cbHeight$, the number of luma coding subblocks in horizontal direction $numSbX$ and in vertical direction $numSbY$, the chroma motion vectors $mvC[xSbIdx][ySbIdx]$ with $xSbIdx = 0..numSbX - 1$, and $ySbIdx = 0..numSbY - 1$ and the variable $cIdx$ set equal to 1 as inputs, and the ibc prediction samples ($predSamples$) that are an $(cbWidth / 2) \times (cbHeight / 2)$ array $predSamples_{Cb}$ of prediction chroma samples for the chroma components Cb as outputs.
 - The decoding process for ibc blocks as specified in clause 8.6.3.1 is invoked with the luma coding block location (xCb, yCb) , the luma coding block width $cbWidth$ and the luma coding block height $cbHeight$, the number of luma coding subblocks in horizontal direction $numSbX$ and in vertical direction $numSbY$, the chroma motion vectors $mvC[xSbIdx][ySbIdx]$ with $xSbIdx = 0..numSbX - 1$, and $ySbIdx = 0..numSbY - 1$ and the variable $cIdx$ set equal to 2 as inputs, and the ibc prediction samples ($predSamples$) that are an $(cbWidth / 2) \times (cbHeight / 2)$ array $predSamples_{Cr}$ of prediction chroma samples for the chroma components Cr as outputs.
3. The variables $NumSbX[xCb][yCb]$ and $NumSbY[xCb][yCb]$ are set equal to $numSbX$ and $numSbY$, respectively.
 4. The residual samples of the current coding unit are derived as follows:
 - **WHEN TREETYPE IS EQUAL TO SINGLE_TREE OR TREETYPE IS EQUAL TO DUAL_TREE_LUMA**, the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in clause 8.5.8 is invoked with the location $(xTb0, yTb0)$ set equal to the luma location (xCb, yCb) , the width $nTbW$ set equal to the luma coding block width $cbWidth$, the height $nTbH$ set equal to the luma coding block height $cbHeight$ and the variable $cIdx$ set equal to 0 as inputs, and the array $resSamples_L$ as output.
 - When $treeType$ is equal to **SINGLE_TREE OR TREETYPE IS EQUAL TO DUAL_TREE_CHROMA**, the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in clause 8.5.8 is invoked with the location $(xTb0, yTb0)$ set equal to the chroma location $(xCb / 2, yCb / 2)$, the width $nTbW$ set equal

to the chroma coding block width $cbWidth / 2$, the height $nTbH$ set equal to the chroma coding block height $cbHeight / 2$ and the variable $cIdx$ set equal to 1 as inputs, and the array $resSamples_{Cb}$ as output.

- When $treeType$ is equal to **SINGLE_TREE OR TREETYPE IS EQUAL TO DUAL_TREE_CHROMA**, the decoding process for the residual signal of coding blocks coded in inter prediction mode as specified in clause 8.5.8 is invoked with the location $(xTb0, yTb0)$ set equal to the chroma location $(xCb / 2, yCb / 2)$, the width $nTbW$ set equal to the chroma coding block width $cbWidth / 2$, the height $nTbH$ set equal to the chroma coding block height $cbHeight / 2$ and the variable $cIdx$ set equal to 2 as inputs, and the array $resSamples_{Cr}$ as output.

5. The reconstructed samples of the current coding unit are derived as follows:

- **WHEN TREETYPE IS EQUAL TO SINGLE_TREE OR TREETYPE IS EQUAL TO DUAL_TREE_LUMA**, the picture reconstruction process for a colour component as specified in clause 8.7.5 is invoked with the block location (xB, yB) set equal to (xCb, yCb) , the block width $bWidth$ set equal to $cbWidth$, the block height $bHeight$ set equal to $cbHeight$, the variable $cIdx$ set equal to 0, the $(cbWidth) \times (cbHeight)$ array $predSamples$ set equal to $predSamples_L$ and the $(cbWidth) \times (cbHeight)$ array $resSamples$ set equal to $resSamples_L$ as inputs, and the output is a modified reconstructed picture before in-loop filtering.
- When $treeType$ is equal to **SINGLE_TREE OR TREETYPE IS EQUAL TO DUAL_TREE_CHROMA**, the picture reconstruction process for a colour component as specified in clause 8.7.5 is invoked with the block location (xB, yB) set equal to $(xCb / 2, yCb / 2)$, the block width $bWidth$ set equal to $cbWidth / 2$, the block height $bHeight$ set equal to $cbHeight / 2$, the variable $cIdx$ set equal to 1, the $(cbWidth / 2) \times (cbHeight / 2)$ array $predSamples$ set equal to $predSamples_{Cb}$ and the $(cbWidth / 2) \times (cbHeight / 2)$ array $resSamples$ set equal to $resSamples_{Cb}$ as inputs, and the output is a modified reconstructed picture before in-loop filtering.
- When $treeType$ is equal to **SINGLE_TREE OR TREETYPE IS EQUAL TO DUAL_TREE_CHROMA**, the picture reconstruction process for a colour component as specified in clause 8.7.5 is invoked with the block location (xB, yB) set equal to $(xCb / 2, yCb / 2)$, the block width $bWidth$ set equal to $cbWidth / 2$, the block height $bHeight$ set equal to $cbHeight / 2$, the variable $cIdx$ set equal to 2, the $(cbWidth / 2) \times (cbHeight / 2)$ array $predSamples$ set equal to $predSamples_{Cr}$ and the $(cbWidth / 2) \times (cbHeight / 2)$ array $resSamples$ set equal to $resSamples_{Cr}$ as inputs, and the output is a modified reconstructed picture before in-loop filtering.

CLAIMS

1. A method of processing video data, comprising:

determining, for a conversion between a current video block in a video region of a video data and a bitstream of the video data, whether to use a transform skip mode for the current video block, wherein in response to using the transform skip mode, a transform is skipped on a prediction residual between the current video block and a reference video block;

selecting, in response to using the transform skip mode, a context of a sign flag for a current coefficient of the current video block according to one or more sign flags of one or more neighboring coefficients in the current video block;

determining whether to enable or disable a cross-component linear model prediction mode on the current video block based on whether the current video block is in a dual tree structure in which a luma component and at least one chroma component have separate partition tree structures, wherein in response to the cross-component linear model prediction mode enabled for the current video block, chroma samples of the current video block are predicted based on reconstructed luma samples of the video region by using a linear model; and

performing the conversion based on the selecting and the determining.

2. The method of claim 1, wherein the one or more neighboring coefficients in the current video block include at least one of a left neighbor coefficient or an above neighbor coefficient.

3. The method of claim 1 or claim 2, wherein the one or more neighboring coefficients in the current video block include: a left neighbor coefficient and an above neighbor coefficient.

4. The method of any one of claims 1-3, wherein the context of the sign flag for the current coefficient is based on the one or more sign flags of the one or more neighboring coefficients in accordance with a following rule:

if sign flags of a pair of the one or more neighboring coefficients are both negative, the rule specifies selecting a first context.

5. The method of any one of claims 1-3, wherein the context of the sign flag for the current coefficient is based on the one or more sign flags of the one or more neighboring coefficients in accordance with a following rule:

if one sign flag of a pair of the one or more neighboring coefficients is negative and another sign flag of a pair of the one or more neighboring coefficients is positive, the rule specifies selecting a second context.

6. The method of any one of claims 1-3, wherein the context of the sign flag for the current coefficient is based on the one or more sign flags of the one or more neighboring coefficients in accordance with a following rule:

if sign flags of a pair of the one or more neighboring coefficients are both positive, the rule specifies selecting a third context.

7. The method of any one of claims 1-6, wherein a relationship between the context of the sign flag of the current coefficient and one or more sign flags of the one or more neighboring coefficients is expressed as:

$C = (L + A)$, where C is a context identifier of the context of the sign flag of the current coefficient, L is a sign flag of a left neighbor coefficient of the current coefficient, and A is a sign flag of an above neighbor coefficient of the current coefficient.

8. The method of any one of claims 1-7, wherein the context of the sign flag for the current coefficient is further based on the position of the current coefficient.

9. The method of any one of claims 1-8, wherein a first syntax element indicating whether the transform skip mode is applied to the current video block is conditionally included in the bitstream.
10. The method of claim 9, wherein whether the first syntax element is included in the bitstream is based on a block dimension of the current video block.
11. The method of claim 9, wherein whether the first syntax element is included in the bitstream is based on a coding tree structure of the current video block.
12. The method of claim 11, wherein the first syntax element is included in the bitstream under conditions including a condition that the current video block is in a single tree structure.
13. The method of claim 11, wherein the first syntax element is included in the bitstream under conditions including a condition that the current video block is in a dual tree luma structure.
14. The method of claim 11, wherein the first syntax element is included in the bitstream under conditions including a condition that the current video block is not in a dual tree chroma structure.
15. The method of any one of claims 1-14, wherein a second syntax element indicating a maximum allowed block size used for the transform skip mode is included in the bitstream.

16. The method of any one of claims 1-15, wherein the conversion includes encoding the current video block into the bitstream.

17. The method of any one of claims 1-15, wherein the conversion includes decoding the current video block from the bitstream.

18. An apparatus for processing video data comprising a processor and a non-transitory memory with instructions thereon, wherein the instructions upon execution by the processor, cause the processor to:

determine, for a conversion between a current video block in a video region of a video data and a bitstream of the video data, whether to use a transform skip mode for the current video block, wherein in response to using the transform skip mode, a transform is skipped on a prediction residual between the current video block and a reference video block;

select, in response to using the transform skip mode, a context of a sign flag for a current coefficient of the current video block according to one or more sign flags of one or more neighboring coefficients in the current video block;

determine whether to enable or disable a cross-component linear model prediction mode on the current video block based on whether the current video block is in a dual tree structure in which a luma component and at least one chroma component have separate partition tree structures, wherein in response to the cross-component linear model prediction mode enabled for the current video block, chroma samples of the current video block are predicted based on reconstructed luma samples of the video region by using a linear model; and

perform the conversion based on the selecting and the determining.

19. A non-transitory computer-readable storage medium storing instructions that cause a processor to:

determine, for a conversion between a current video block in a video region of a video data and a bitstream of the video data, whether to use a transform skip mode for the current video block, wherein in response to using the transform skip mode, a transform is skipped on a prediction residual between the current video block and a reference video block;

select, in response to using the transform skip mode, a context of a sign flag for a current coefficient of the current video block according to one or more sign flags of one or more neighboring coefficients in the current video block;

determine whether to enable or disable a cross-component linear model prediction mode on the current video block based on whether the current video block is in a dual tree structure in which a luma component and at least one chroma component have separate partition tree structures, wherein in response to the cross-component linear model prediction mode enabled for the current video block, chroma samples of the current video block are predicted based on reconstructed luma samples of the video region by using a linear model; and

perform the conversion based on the selecting and the determining.

20. A non-transitory computer-readable recording medium storing a bitstream of a video which is generated by a method performed by a video processing apparatus, wherein the method comprises:

determining whether to use a transform skip mode for a current video block in a video region of a video data, wherein in response to using the transform skip mode, a transform is skipped on a prediction residual between the current video block and a reference video block;

selecting, in response to using the transform skip mode, a context of a sign flag for a current coefficient of the current video block according to one or more sign flags of one or more neighboring coefficients in the current video block;

determining whether to enable or disable a cross-component linear model prediction mode on the current video block based on whether the current video block is in a dual tree structure in which a luma component and at least one chroma component have separate partition tree structures, wherein in response to the cross-component linear model prediction mode enabled for the current video block, chroma samples of the current video block are predicted based on reconstructed luma samples of the video region by using a linear model; and

generating the bitstream of the video data based on the selecting and the determining.