METHOD AND APPARATUS FOR COLOR TRANSFORM IN VVC

INCORPORATION BY REFERENCE

[0001] This present disclosure claims the benefit of priority of U.S. Non-Provisional Application No. 16/817,028, "METHOD AND APPARATUS FOR COLOR TRANSFORM IN VVC", filed on March 12, 2020, which claims the benefit of priority to U.S. Provisional Application No. 62/817,500, "COLOR TRANSFORM IN VVC" filed on March 12, 2019. The entire disclosures of the prior applications are hereby incorporated by reference in their entirety.

TECHNICAL FIELD

[0002] The present disclosure describes embodiments generally related to video coding.

BACKGROUND

[0003] The background description provided herein is for the purpose of generally presenting the context of the disclosure. Work of the presently named inventors, to the extent the work is described in this background section, as well as aspects of the description that may not otherwise qualify as prior art at the time of filing, are neither expressly nor impliedly admitted as prior art against the present disclosure.

[0004] Video coding and decoding can be performed using inter-picture prediction with motion compensation. Uncompressed digital video can include a series of pictures, each picture having a spatial dimension of, for example, 1920 x 1080 luminance samples and associated chrominance samples. The series of pictures can have a fixed or variable picture rate (informally also known as frame rate), of, for example 60 pictures per second or 60 Hz. Uncompressed video has significant bitrate requirements. For example, 1080p60 4:2:0 video at 8 bit per sample (1920x1080 luminance sample resolution at 60 Hz frame rate) requires close to 1.5 Gbit/s bandwidth. An hour of such video requires more than 600 GBytes of storage space.

[0005] One purpose of video coding and decoding can be the reduction of redundancy in the input video signal, through compression. Compression can help reduce the aforementioned bandwidth or storage space requirements, in some cases by two orders of magnitude or more. Both lossless and lossy compression, as well as a combination thereof

can be employed. Lossless compression refers to techniques where an exact copy of the original signal can be reconstructed from the compressed original signal. When using lossy compression, the reconstructed signal may not be identical to the original signal, but the distortion between original and reconstructed signals is small enough to make the reconstructed signal useful for the intended application. In the case of video, lossy compression is widely employed. The amount of distortion tolerated depends on the application; for example, users of certain consumer streaming applications may tolerate higher distortion than users of television distribution applications. The compression ratio achievable can reflect that: higher allowable/tolerable distortion can yield higher compression ratios.

[0006] A video encoder and decoder can utilize techniques from several broad categories, including, for example, motion compensation, transform, quantization, and entropy coding.

[0007] Video codec technologies can include techniques known as intra coding. In intra coding, sample values are represented without reference to samples or other data from previously reconstructed reference pictures. In some video codecs, the picture is spatially subdivided into blocks of samples. When all blocks of samples are coded in intra mode, that picture can be an intra picture. Intra pictures and their derivations such as independent decoder refresh pictures, can be used to reset the decoder state and can, therefore, be used as the first picture in a coded video bitstream and a video session, or as a still image. The samples of an intra block can be exposed to a transform, and the transform coefficients can be quantized before entropy coding. Intra prediction can be a technique that minimizes sample values in the pre-transform domain. In some cases, the smaller the DC value after a transform is, and the smaller the AC coefficients are, the fewer the bits that are required at a given quantization step size to represent the block after entropy coding.

[0008] Traditional intra coding such as known from, for example MPEG-2 generation coding technologies, does not use intra prediction. However, some newer video compression technologies include techniques that attempt, from, for example, surrounding sample data and/or metadata obtained during the encoding/decoding of spatially neighboring, and preceding in decoding order, blocks of data. Such techniques are henceforth called "intra prediction" techniques. Note that in at least some cases, intra prediction is only using reference data from the current picture under reconstruction and not from reference pictures.

[0009] There can be many different forms of intra prediction. When more than one of such techniques can be used in a given video coding technology, the technique in use can be

coded in an intra prediction mode. In certain cases, modes can have submodes and/or parameters, and those can be coded individually or included in the mode codeword. Which codeword to use for a given mode/submode/parameter combination can have an impact in the coding efficiency gain through intra prediction, and so can the entropy coding technology used to translate the codewords into a bitstream.

[0010] A certain mode of intra prediction was introduced with H.264, refined in H.265, and further refined in newer coding technologies such as joint exploration model (JEM), versatile video coding (VVC), and benchmark set (BMS). A predictor block can be formed using neighboring sample values belonging to already available samples. Sample values of neighboring samples are copied into the predictor block according to a direction. A reference to the direction in use can be coded in the bitstream or may itself be predicted.

[0011] Referring to FIG. 1A, depicted in the lower right is a subset of nine predictor directions known from H.265's 33 possible predictor directions (corresponding to the 33 angular modes of the 35 intra modes). The point where the arrows converge (101) represents the sample being predicted. The arrows represent the direction from which the sample is being predicted. For example, arrow (102) indicates that sample (101) is predicted from a sample or samples to the upper right, at a 45 degree angle from the horizontal. Similarly, arrow (103) indicates that sample (101) is predicted from a sample or samples to the lower left of sample (101), in a 22.5 degree angle from the horizontal.

[0012] Still referring to FIG. 1A, on the top left there is depicted a square block (104) of 4 x 4 samples (indicated by a dashed, boldface line). The square block (104) includes 16 samples, each labelled with an "S", its position in the Y dimension (e.g., row index) and its position in the X dimension (e.g., column index). For example, sample S21 is the second sample in the Y dimension (from the top) and the first (from the left) sample in the X dimension. Similarly, sample S44 is the fourth sample in block (104) in both the Y and X dimensions. As the block is 4 x 4 samples in size, S44 is at the bottom right. Further shown are reference samples that follow a similar numbering scheme. A reference sample is labelled with an R, its Y position (e.g., row index) and X position (column index) relative to block (104). In both H.264 and H.265, prediction samples neighbor the block under reconstruction; therefore no negative values need to be used.

[0013] Intra picture prediction can work by copying reference sample values from the neighboring samples as appropriated by the signaled prediction direction. For example, assume the coded video bitstream includes signaling that, for this block, indicates a prediction direction consistent with arrow (102)—that is, samples are predicted from a

prediction sample or samples to the upper right, at a 45 degree angle from the horizontal. In that case, samples S41, S32, S23, and S14 are predicted from the same reference sample R05. Sample S44 is then predicted from reference sample R08.

[0014] In certain cases, the values of multiple reference samples may be combined, for example through interpolation, in order to calculate a reference sample; especially when the directions are not evenly divisible by 45 degrees.

[0015] The number of possible directions has increased as video coding technology has developed. In H.264 (year 2003), nine different direction could be represented. That increased to 33 in H.265 (year 2013), and JEM/VVC/BMS, at the time of disclosure, can support up to 65 directions. Experiments have been conducted to identify the most likely directions, and certain techniques in the entropy coding are used to represent those likely directions in a small number of bits, accepting a certain penalty for less likely directions. Further, the directions themselves can sometimes be predicted from neighboring directions used in neighboring, already decoded, blocks.

[0016] The intra prediction modes used in HEVC are illustrated in FIG. 1B. In HEVC, there are total 35 intra prediction modes, among which mode 10 is horizontal mode, mode 26 is vertical mode, and mode 2, mode 18 and mode 34 are diagonal modes. The intra prediction modes are signalled by three most probable modes (MPMs) and 32 remaining modes.

[0017] FIG. 1C illustrates the intra prediction modes used in VVC. In VVC, there are total 95 intra prediction modes as shown in FIG. 1C, where mode 18 is the horizontal mode, mode 50 is the vertical mode, and mode 2, mode 34 and mode 66 are diagonal modes. Modes -1 ~ -14 and Modes 67 ~ 80 are called Wide-Angle Intra Prediction (WAIP) modes.

[0018] The mapping of intra prediction directions bits in the coded video bitstream that represent the direction can be different from video coding technology to video coding technology; and can range, for example, from simple direct mappings of prediction direction to intra prediction mode, to codewords, to complex adaptive schemes involving MPMs, and similar techniques. In all cases, however, there can be certain directions that are statistically less likely to occur in video content than certain other directions. As the goal of video compression is the reduction of redundancy, those less likely directions will, in a well working video coding technology, be represented by a larger number of bits than more likely directions.

[0019] Video coding and decoding can be performed using inter-picture prediction with motion compensation. Uncompressed digital video can include a series of pictures, each

picture having a spatial dimension of, for example, 1920 x 1080 luminance samples and associated chrominance samples. The series of pictures can have a fixed or variable picture rate (informally also known as frame rate), of, for example 60 pictures per second or 60 Hz. Uncompressed video has significant bitrate requirements. For example, 1080p60 4:2:0 video at 8 bit per sample (1920x1080 luminance sample resolution at 60 Hz frame rate) requires close to 1.5 Gbit/s bandwidth. An hour of such video requires more than 600 GBytes of storage space.

[0020] Motion compensation can be a lossy compression technique and can relate to techniques where a block of sample data from a previously reconstructed picture or part thereof (reference picture), after being spatially shifted in a direction indicated by a motion vector (MV henceforth), is used for the prediction of a newly reconstructed picture or picture part. In some cases, the reference picture can be the same as the picture currently under reconstruction. MVs can have two dimensions X and Y, or three dimensions, the third being an indication of the reference picture in use (the latter, indirectly, can be a time dimension).

[0021] In some video compression techniques, an MV applicable to a certain area of sample data can be predicted from other MVs, for example from those related to another area of sample data spatially adjacent to the area under reconstruction, and preceding that MV in decoding order. Doing so can substantially reduce the amount of data required for coding the MV, thereby removing redundancy and increasing compression. MV prediction can work effectively, for example, because when coding an input video signal derived from a camera (known as natural video) there is a statistical likelihood that areas larger than the area to which a single MV is applicable move in a similar direction and, therefore, can in some cases be predicted using a similar motion vector derived from MVs of neighboring area. That results in the MV found for a given area to be similar or the same as the MV predicted from the surrounding MVs, and that in turn can be represented, after entropy coding, in a smaller number of bits than what would be used if coding the MV directly. In some cases, MV prediction can be an example of lossless compression of a signal (namely: the MVs) derived from the original signal (namely: the sample stream). In other cases, MV prediction itself can be lossy, for example because of rounding errors when calculating a predictor from several surrounding MVs.

[0022] Various MV prediction mechanisms are described in H.265/HEVC (ITU-T Rec. H.265, "High Efficiency Video Coding", December 2016). Out of the many MV prediction mechanisms that H.265 offers, described here is a technique henceforth referred to as "spatial merge".

[0023] Referring to FIG. 1D, a current block (101) comprises samples that have been found by the encoder during the motion search process to be predictable from a previous block of the same size that has been spatially shifted. Instead of coding that MV directly, the MV can be derived from metadata associated with one or more reference pictures, for example from the most recent (in decoding order) reference picture, using the MV associated with either one of five surrounding samples, denoted A0, A1, and B0, B1, B2 (102 through 106, respectively). In H.265, the MV prediction can use predictors from the same reference picture that the neighboring block is using. The order of forming a candidate list may be A0 → B0 → B1 → A1 → B2.

SUMMARY

[0024] According to an exemplary embodiment, a method of video decoding performed in a video decoder. The method including receiving a coded video bitstream including a current picture. The method further including, performing an inverse quantization on a current block included in the current picture. The method further including performing, after performing the inverse quantization, an inverse transform on the current block. The method further including performing a prediction process on the current block after performing the inverse transform. The method further including, after performing the prediction process on the current block, determining whether a predetermined condition is satisfied. The method further including, in response to determining that the predetermined condition is met, performing an inverse color transform on the current block.

[0025] According to an exemplary embodiment, a video decoder for video decoding, including processing circuitry configured to: receive a coded video bitstream including a current picture. The processing circuitry is further configured to perform an inverse quantization on a current block included in the current picture. The processing circuitry is further configured to perform, after performance of the inverse quantization, an inverse transform on the current block. The processing circuitry is further configured to perform a prediction process on the current block after performance of the inverse transform. The processing circuitry is further configured to, after performance of the prediction process on the current block, determine whether a predetermined condition is satisfied. The processing circuitry is further configured to, in response to a determination that the predetermined condition is satisfied, perform an inverse color transform on the current block.

[0026] According to an exemplary embodiment, a non-transitory computer readable medium having instructions stored therein, which when executed by a processor in a video

decoder causes the video decoder to execute a method including receiving a coded video bitstream including a current picture. The method further including, performing an inverse quantization on a current block included in the current picture. The method further including performing, after performing the inverse quantization, an inverse transform on the current block. The method further including performing a prediction process on the current block after performing the inverse transform. The method further including, after performing the prediction process on the current block, determining whether a predetermined condition is satisfied. The method further including, in response to determining that the predetermined condition is met, performing an inverse color transform on the current block

BRIEF DESCRIPTION OF THE DRAWINGS

[0027] Further features, the nature, and various advantages of the disclosed subject matter will be more apparent from the following detailed description and the accompanying drawings in which:

[0028] FIG. 1A is a schematic illustration of an exemplary subset of intra prediction modes;

[0029] FIG. 1B is an illustration of exemplary intra prediction directions.

[0030] FIG. 1C is an illustration of exemplary intra prediction directions.

[0031] FIG. 1D is a schematic illustration of a current block and its surrounding spatial merge candidates in one example.

[0032] FIG. 2 is a schematic illustration of a simplified block diagram of a communication system (200) in accordance with an embodiment.

[0033] FIG. 3 is a schematic illustration of a simplified block diagram of a communication system (300) in accordance with an embodiment.

[0034] FIG. 4 is a schematic illustration of a simplified block diagram of a decoder in accordance with an embodiment.

[0035] FIG. 5 is a schematic illustration of a simplified block diagram of an encoder in accordance with an embodiment.

[0036] FIG. 6 shows a block diagram of an encoder in accordance with another embodiment.

[0037] FIG. 7 shows a block diagram of a decoder in accordance with another embodiment.

[0038] FIG. 8A shows block partitioning in accordance with an embodiment.

[0039] FIG. 8B shows a block partitioning tree in accordance with an embodiment.

[0040] FIG. 9A shows a vertical center-side ternary tree partitioning in accordance with an embodiment.

[0041] FIG. 9B shows a horizontal center-side ternary tree partitioning in accordance with an embodiment.

[0042] FIGS. 10A-10D illustrate different chroma formats in accordance with various embodiments.

[0043] FIG. 11 illustrates an example encoder in accordance with an embodiment.

[0044] FIG. 12 illustrates an example decoder in accordance with an embodiment.

[0045] FIG. 13 illustrates a straight line between a minimum and maximum luma value in accordance with an embodiment.

[0046] FIGs. 14A and 14B illustrate locations of samples used for derivation of α and β in LT_CCLM in accordance with an embodiment.

[0047] FIGs. 15A and 15B illustrate locations of the samples used for the derivation of α and β in T_CCLM in accordance with an embodiment.

[0048] FIGs. 16A and 16B illustrate locations of the samples used for the derivation of α and β in L_CCLM in accordance with an embodiment.

[0049] FIG. 17 illustrates an example of classifying neighboring samples into two groups in accordance with an embodiment.

[0050] FIG. 18 is a schematic illustration of an encoder and a decoder in accordance with an embodiment.

[0051] FIG. 19 is an illustration of an embodiment of a process performed by an encoder.

[0052] FIG. 20 is an illustration of an embodiment of a process performed by a decoder.

[0053] FIG. 21 is a schematic illustration of a computer system in accordance with an embodiment of the present disclosure.

DETAILED DESCRIPTION OF EMBODIMENTS

[0054] FIG. 2 illustrates a simplified block diagram of a communication system (200) according to an embodiment of the present disclosure. The communication system (200) includes a plurality of terminal devices that can communicate with each other, via, for example, a network (250). For example, the communication system (200) includes a first pair of terminal devices (210) and (220) interconnected via the network (250). In the FIG. 2 example, the first pair of terminal devices (210) and (220) performs unidirectional

transmission of data. For example, the terminal device (210) may code video data (e.g., a stream of video pictures that are captured by the terminal device (210)) for transmission to the other terminal device (220) via the network (250). The encoded video data can be transmitted in the form of one or more coded video bitstreams. The terminal device (220) may receive the coded video data from the network (250), decode the coded video data to recover the video pictures and display video pictures according to the recovered video data. Unidirectional data transmission may be common in media serving applications and the like.

[0055] In another example, the communication system (200) includes a second pair of terminal devices (230) and (240) that performs bidirectional transmission of coded video data that may occur, for example, during videoconferencing. For bidirectional transmission of data, in an example, each terminal device of the terminal devices (230) and (240) may code video data (e.g., a stream of video pictures that are captured by the terminal device) for transmission to the other terminal device of the terminal devices (230) and (240) via the network (250). Each terminal device of the terminal devices (230) and (240) also may receive the coded video data transmitted by the other terminal device of the terminal devices (230) and (240), and may decode the coded video data to recover the video pictures and may display video pictures at an accessible display device according to the recovered video data.

[0056] In the FIG. 2 example, the terminal devices (210), (220), (230) and (240) may be illustrated as servers, personal computers and smart phones but the principles of the present disclosure may be not so limited. Embodiments of the present disclosure find application with laptop computers, tablet computers, media players and/or dedicated video conferencing equipment. The network (250) represents any number of networks that convey coded video data among the terminal devices (210), (220), (230) and (240), including for example wireline (wired) and/or wireless communication networks. The communication network (250) may exchange data in circuit-switched and/or packet-switched channels. Representative networks include telecommunications networks, local area networks, wide area networks and/or the Internet. For the purposes of the present discussion, the architecture and topology of the network (250) may be immaterial to the operation of the present disclosure unless explained herein below.

[0057] FIG. 3 illustrates, as an example for an application for the disclosed subject matter, the placement of a video encoder and a video decoder in a streaming environment. The disclosed subject matter can be equally applicable to other video enabled applications, including, for example, video conferencing, digital TV, storing of compressed video on digital media including CD, DVD, memory stick and the like, and so on.

[0058] A streaming system may include a capture subsystem (313), that can include a video source (301), for example a digital camera, creating for example a stream of video pictures (302) that are uncompressed. In an example, the stream of video pictures (302) includes samples that are taken by the digital camera. The stream of video pictures (302), depicted as a bold line to emphasize a high data volume when compared to encoded video data (304) (or coded video bitstreams), can be processed by an electronic device (320) that includes a video encoder (303) coupled to the video source (301). The video encoder (303) can include hardware, software, or a combination thereof to enable or implement aspects of the disclosed subject matter as described in more detail below. The encoded video data (304) (or encoded video bitstream (304)), depicted as a thin line to emphasize the lower data volume when compared to the stream of video pictures (302), can be stored on a streaming server (305) for future use. One or more streaming client subsystems, such as client subsystems (306) and (308) in FIG. 3 can access the streaming server (305) to retrieve copies (307) and (309) of the encoded video data (304). A client subsystem (306) can include a video decoder (310), for example, in an electronic device (330). The video decoder (310) decodes the incoming copy (307) of the encoded video data and creates an outgoing stream of video pictures (311) that can be rendered on a display (312) (e.g., display screen) or other rendering device (not depicted). In some streaming systems, the encoded video data (304), (307), and (309) (e.g., video bitstreams) can be encoded according to certain video coding/compression standards. Examples of those standards include ITU-T Recommendation H.265. In an example, a video coding standard under development is informally known as Versatile Video Coding (VVC). The disclosed subject matter may be used in the context of VVC.

[0059] It is noted that the electronic devices (320) and (330) can include other components (not shown). For example, the electronic device (320) can include a video decoder (not shown) and the electronic device (330) can include a video encoder (not shown) as well.

[0060] FIG. 4 shows a block diagram of a video decoder (410) according to an embodiment of the present disclosure. The video decoder (410) can be included in an electronic device (430). The electronic device (430) can include a receiver (431) (e.g., receiving circuitry). The video decoder (410) can be used in the place of the video decoder (310) in the FIG. 3 example.

[0061] The receiver (431) may receive one or more coded video sequences to be decoded by the video decoder (410); in the same or another embodiment, one coded video

sequence at a time, where the decoding of each coded video sequence is independent from other coded video sequences. The coded video sequence may be received from a channel (401), which may be a hardware/software link to a storage device which stores the encoded video data. The receiver (431) may receive the encoded video data with other data, for example, coded audio data and/or ancillary data streams, that may be forwarded to their respective using entities (not depicted). The receiver (431) may separate the coded video sequence from the other data. To combat network jitter, a buffer memory (415) may be coupled in between the receiver (431) and an entropy decoder / parser (420) ("parser (420)" henceforth). In certain applications, the buffer memory (415) is part of the video decoder (410). In others, it can be outside of the video decoder (410) (not depicted). In still others, there can be a buffer memory (not depicted) outside of the video decoder (410), for example to combat network jitter, and in addition another buffer memory (415) inside the video decoder (410), for example to handle playout timing. When the receiver (431) is receiving data from a store/forward device of sufficient bandwidth and controllability, or from an isosynchronous network, the buffer memory (415) may not be needed, or can be small. For use on best effort packet networks such as the Internet, the buffer memory (415) may be required, can be comparatively large and can be advantageously of adaptive size, and may at least partially be implemented in an operating system or similar elements (not depicted) outside of the video decoder (410).

[0062] The video decoder (410) may include the parser (420) to reconstruct symbols (421) from the coded video sequence. Categories of those symbols include information used to manage operation of the video decoder (410), and potentially information to control a rendering device such as a render device (412) (e.g., a display screen) that is not an integral part of the electronic device (430) but can be coupled to the electronic device (430), as was shown in FIG. 4. The control information for the rendering device(s) may be in the form of Supplemental Enhancement Information (SEI messages) or Video Usability Information (VUI) parameter set fragments (not depicted). The parser (420) may parse / entropy-decode the coded video sequence that is received. The coding of the coded video sequence can be in accordance with a video coding technology or standard, and can follow various principles, including variable length coding, Huffman coding, arithmetic coding with or without context sensitivity, and so forth. The parser (420) may extract from the coded video sequence, a set of subgroup parameters for at least one of the subgroups of pixels in the video decoder, based upon at least one parameter corresponding to the group. Subgroups can include Groups of Pictures (GOPs), pictures, tiles, slices, macroblocks, Coding Units (CUs), blocks, Transform

Units (TUs), Prediction Units (PUs) and so forth. The parser (420) may also extract from the coded video sequence information such as transform coefficients, quantizer parameter values, motion vectors, and so forth.

[0063] The parser (420) may perform an entropy decoding / parsing operation on the video sequence received from the buffer memory (415), so as to create symbols (421).

[0064] Reconstruction of the symbols (421) can involve multiple different units depending on the type of the coded video picture or parts thereof (such as: inter and intra picture, inter and intra block), and other factors. Which units are involved, and how, can be controlled by the subgroup control information that was parsed from the coded video sequence by the parser (420). The flow of such subgroup control information between the parser (420) and the multiple units below is not depicted for clarity.

[0065] Beyond the functional blocks already mentioned, the video decoder (410) can be conceptually subdivided into a number of functional units as described below. In a practical implementation operating under commercial constraints, many of these units interact closely with each other and can, at least partly, be integrated into each other. However, for the purpose of describing the disclosed subject matter, the conceptual subdivision into the functional units below is appropriate.

[0066] A first unit is the scaler / inverse transform unit (451). The scaler / inverse transform unit (451) receives a quantized transform coefficient as well as control information, including which transform to use, block size, quantization factor, quantization scaling matrices, etc. as symbol(s) (421) from the parser (420). The scaler / inverse transform unit (451) can output blocks comprising sample values, that can be input into aggregator (455).

[0067] In some cases, the output samples of the scaler / inverse transform (451) can pertain to an intra coded block; that is: a block that is not using predictive information from previously reconstructed pictures, but can use predictive information from previously reconstructed parts of the current picture. Such predictive information can be provided by an intra picture prediction unit (452). In some cases, the intra picture prediction unit (452) generates a block of the same size and shape of the block under reconstruction, using surrounding already reconstructed information fetched from the current picture buffer (458). The current picture buffer (458) buffers, for example, partly reconstructed current picture and/or fully reconstructed current picture. The aggregator (455), in some cases, adds, on a per sample basis, the prediction information the intra prediction unit (452) has generated to the output sample information as provided by the scaler / inverse transform unit (451).

[0068] In other cases, the output samples of the scaler / inverse transform unit (451) can pertain to an inter coded, and potentially motion compensated block. In such a case, a motion compensation prediction unit (453) can access reference picture memory (457) to fetch samples used for prediction. After motion compensating the fetched samples in accordance with the symbols (421) pertaining to the block, these samples can be added by the aggregator (455) to the output of the scaler / inverse transform unit (451) (in this case called the residual samples or residual signal) so as to generate output sample information. The addresses within the reference picture memory (457) from where the motion compensation prediction unit (453) fetches prediction samples can be controlled by motion vectors, available to the motion compensation prediction unit (453) in the form of symbols (421) that can have, for example X, Y, and reference picture components. Motion compensation also can include interpolation of sample values as fetched from the reference picture memory (457) when sub-sample exact motion vectors are in use, motion vector prediction mechanisms, and so forth.

[0069] The output samples of the aggregator (455) can be subject to various loop filtering techniques in the loop filter unit (456). Video compression technologies can include in-loop filter technologies that are controlled by parameters included in the coded video sequence (also referred to as coded video bitstream) and made available to the loop filter unit (456) as symbols (421) from the parser (420), but can also be responsive to meta-information obtained during the decoding of previous (in decoding order) parts of the coded picture or coded video sequence, as well as responsive to previously reconstructed and loop-filtered sample values.

[0070] The output of the loop filter unit (456) can be a sample stream that can be output to the render device (412) as well as stored in the reference picture memory (457) for use in future inter-picture prediction.

[0071] Certain coded pictures, once fully reconstructed, can be used as reference pictures for future prediction. For example, once a coded picture corresponding to a current picture is fully reconstructed and the coded picture has been identified as a reference picture (by, for example, the parser (420)), the current picture buffer (458) can become a part of the reference picture memory (457), and a fresh current picture buffer can be reallocated before commencing the reconstruction of the following coded picture.

[0072] The video decoder (410) may perform decoding operations according to a predetermined video compression technology in a standard, such as ITU-T Rec. H.265. The coded video sequence may conform to a syntax specified by the video compression

technology or standard being used, in the sense that the coded video sequence adheres to both the syntax of the video compression technology or standard and the profiles as documented in the video compression technology or standard. Specifically, a profile can select certain tools as the only tools available for use under that profile from all the tools available in the video compression technology or standard. Also necessary for compliance can be that the complexity of the coded video sequence is within bounds as defined by the level of the video compression technology or standard. In some cases, levels restrict the maximum picture size, maximum frame rate, maximum reconstruction sample rate (measured in, for example megasamples per second), maximum reference picture size, and so on. Limits set by levels can, in some cases, be further restricted through Hypothetical Reference Decoder (HRD) specifications and metadata for HRD buffer management signaled in the coded video sequence.

[0073] In an embodiment, the receiver (431) may receive additional (redundant) data with the encoded video. The additional data may be included as part of the coded video sequence(s). The additional data may be used by the video decoder (410) to properly decode the data and/or to more accurately reconstruct the original video data. Additional data can be in the form of, for example, temporal, spatial, or signal noise ratio (SNR) enhancement layers, redundant slices, redundant pictures, forward error correction codes, and so on.

[0074] FIG. 5 shows a block diagram of a video encoder (503) according to an embodiment of the present disclosure. The video encoder (503) is included in an electronic device (520). The electronic device (520) includes a transmitter (540) (e.g., transmitting circuitry). The video encoder (503) can be used in the place of the video encoder (303) in the FIG. 3 example.

[0075] The video encoder (503) may receive video samples from a video source (501) (that is not part of the electronic device (520) in the FIG. 5 example) that may capture video image(s) to be coded by the video encoder (503). In another example, the video source (501) is a part of the electronic device (520).

[0076] The video source (501) may provide the source video sequence to be coded by the video encoder (503) in the form of a digital video sample stream that can be of any suitable bit depth (for example: 8 bit, 10 bit, 12 bit, ...), any colorspace (for example, BT.601 Y CrCB, RGB, ...), and any suitable sampling structure (for example Y CrCb 4:2:0, Y CrCb 4:4:4). In a media serving system, the video source (501) may be a storage device storing previously prepared video. In a videoconferencing system, the video source (501) may be a camera that captures local image information as a video sequence. Video data may be

provided as a plurality of individual pictures that impart motion when viewed in sequence. The pictures themselves may be organized as a spatial array of pixels, wherein each pixel can comprise one or more samples depending on the sampling structure, color space, etc. in use. A person skilled in the art can readily understand the relationship between pixels and samples. The description below focuses on samples.

[0077] According to an embodiment, the video encoder (503) may code and compress the pictures of the source video sequence into a coded video sequence (543) in real time or under any other time constraints as required by the application. Enforcing appropriate coding speed is one function of a controller (550). In some embodiments, the controller (550) controls other functional units as described below and is functionally coupled to the other functional units. The coupling is not depicted for clarity. Parameters set by the controller (550) can include rate control related parameters (picture skip, quantizer, lambda value of rate-distortion optimization techniques, ...), picture size, group of pictures (GOP) layout, maximum motion vector search range, and so forth. The controller (550) can be configured to have other suitable functions that pertain to the video encoder (503) optimized for a certain system design.

[0078] In some embodiments, the video encoder (503) is configured to operate in a coding loop. As an oversimplified description, in an example, the coding loop can include a source coder (530) (e.g., responsible for creating symbols, such as a symbol stream, based on an input picture to be coded, and a reference picture(s)), and a (local) decoder (533) embedded in the video encoder (503). The decoder (533) reconstructs the symbols to create the sample data in a similar manner as a (remote) decoder also would create (as any compression between symbols and coded video bitstream is lossless in the video compression technologies considered in the disclosed subject matter). The reconstructed sample stream (sample data) is input to the reference picture memory (534). As the decoding of a symbol stream leads to bit-exact results independent of decoder location (local or remote), the content in the reference picture memory (534) is also bit exact between the local encoder and remote encoder. In other words, the prediction part of an encoder "sees" as reference picture samples exactly the same sample values as a decoder would "see" when using prediction during decoding. This fundamental principle of reference picture synchronicity (and resulting drift, if synchronicity cannot be maintained, for example because of channel errors) is used in some related arts as well.

[0079] The operation of the "local" decoder (533) can be the same as of a "remote" decoder, such as the video decoder (410), which has already been described in detail above in

conjunction with FIG. 4. Briefly referring also to FIG. 4, however, as symbols are available and encoding/decoding of symbols to a coded video sequence by an entropy coder (545) and the parser (420) can be lossless, the entropy decoding parts of the video decoder (410), including the buffer memory (415), and parser (420) may not be fully implemented in the local decoder (533).

[0080] An observation that can be made at this point is that any decoder technology except the parsing/entropy decoding that is present in a decoder also necessarily needs to be present, in substantially identical functional form, in a corresponding encoder. For this reason, the disclosed subject matter focuses on decoder operation. The description of encoder technologies can be abbreviated as they are the inverse of the comprehensively described decoder technologies. Only in certain areas a more detail description is required and provided below.

[0081] During operation, in some examples, the source coder (530) may perform motion compensated predictive coding, which codes an input picture predictively with reference to one or more previously-coded picture from the video sequence that were designated as "reference pictures". In this manner, the coding engine (532) codes differences between pixel blocks of an input picture and pixel blocks of reference picture(s) that may be selected as prediction reference(s) to the input picture.

[0082] The local video decoder (533) may decode coded video data of pictures that may be designated as reference pictures, based on symbols created by the source coder (530). Operations of the coding engine (532) may advantageously be lossy processes. When the coded video data may be decoded at a video decoder (not shown in FIG. 5), the reconstructed video sequence typically may be a replica of the source video sequence with some errors. The local video decoder (533) replicates decoding processes that may be performed by the video decoder on reference pictures and may cause reconstructed reference pictures to be stored in the reference picture cache (534). In this manner, the video encoder (503) may store copies of reconstructed reference pictures locally that have common content as the reconstructed reference pictures that will be obtained by a far-end video decoder (absent transmission errors).

[0083] The predictor (535) may perform prediction searches for the coding engine (532). That is, for a new picture to be coded, the predictor (535) may search the reference picture memory (534) for sample data (as candidate reference pixel blocks) or certain metadata such as reference picture motion vectors, block shapes, and so on, that may serve as an appropriate prediction reference for the new pictures. The predictor (535) may operate on

a sample block-by-pixel block basis to find appropriate prediction references. In some cases, as determined by search results obtained by the predictor (535), an input picture may have prediction references drawn from multiple reference pictures stored in the reference picture memory (534).

[0084] The controller (550) may manage coding operations of the source coder (530), including, for example, setting of parameters and subgroup parameters used for encoding the video data.

[0085] Output of all aforementioned functional units may be subjected to entropy coding in the entropy coder (545). The entropy coder (545) translates the symbols as generated by the various functional units into a coded video sequence, by lossless compressing the symbols according to technologies such as Huffman coding, variable length coding, arithmetic coding, and so forth.

[0086] The transmitter (540) may buffer the coded video sequence(s) as created by the entropy coder (545) to prepare for transmission via a communication channel (560), which may be a hardware/software link to a storage device which would store the encoded video data. The transmitter (540) may merge coded video data from the video coder (503) with other data to be transmitted, for example, coded audio data and/or ancillary data streams (sources not shown).

[0087] The controller (550) may manage operation of the video encoder (503). During coding, the controller (550) may assign to each coded picture a certain coded picture type, which may affect the coding techniques that may be applied to the respective picture. For example, pictures often may be assigned as one of the following picture types:

[0088] An Intra Picture (I picture) may be one that may be coded and decoded without using any other picture in the sequence as a source of prediction. Some video codecs allow for different types of intra pictures, including, for example Independent Decoder Refresh ("IDR") Pictures. A person skilled in the art is aware of those variants of I pictures and their respective applications and features.

[0089] A predictive picture (P picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most one motion vector and reference index to predict the sample values of each block.

[0090] A bi-directionally predictive picture (B Picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most two motion vectors and reference indices to predict the sample values of each block. Similarly, multiple-predictive

pictures can use more than two reference pictures and associated metadata for the reconstruction of a single block.

[0091] Source pictures commonly may be subdivided spatially into a plurality of sample blocks (for example, blocks of 4x4, 8x8, 4x8, or 16x16 samples each) and coded on a block-by-block basis. Blocks may be coded predictively with reference to other (already coded) blocks as determined by the coding assignment applied to the blocks' respective pictures. For example, blocks of I pictures may be coded non-predictively or they may be coded predictively with reference to already coded blocks of the same picture (spatial prediction or intra prediction). Pixel blocks of P pictures may be coded predictively, via spatial prediction or via temporal prediction with reference to one previously coded reference picture. Blocks of B pictures may be coded predictively, via spatial prediction or via temporal prediction with reference to one or two previously coded reference pictures.

[0092] The video encoder (503) may perform coding operations according to a predetermined video coding technology or standard, such as ITU-T Rec. H.265. In its operation, the video encoder (503) may perform various compression operations, including predictive coding operations that exploit temporal and spatial redundancies in the input video sequence. The coded video data, therefore, may conform to a syntax specified by the video coding technology or standard being used.

[0093] In an embodiment, the transmitter (540) may transmit additional data with the encoded video. The source coder (530) may include such data as part of the coded video sequence. Additional data may comprise temporal/spatial/SNR enhancement layers, other forms of redundant data such as redundant pictures and slices, SEI messages, VUI parameter set fragments, and so on.

[0094] A video may be captured as a plurality of source pictures (video pictures) in a temporal sequence. Intra-picture prediction (often abbreviated to intra prediction) makes use of spatial correlation in a given picture, and inter-picture prediction makes uses of the (temporal or other) correlation between the pictures. In an example, a specific picture under encoding/decoding, which is referred to as a current picture, is partitioned into blocks. When a block in the current picture is similar to a reference block in a previously coded and still buffered reference picture in the video, the block in the current picture can be coded by a vector that is referred to as a motion vector. The motion vector points to the reference block in the reference picture, and can have a third dimension identifying the reference picture, in case multiple reference pictures are in use.

[0095] In some embodiments, a bi-prediction technique can be used in the inter-picture prediction. According to the bi-prediction technique, two reference pictures, such as a first reference picture and a second reference picture that are both prior in decoding order to the current picture in the video (but may be in the past and future, respectively, in display order) are used. A block in the current picture can be coded by a first motion vector that points to a first reference block in the first reference picture, and a second motion vector that points to a second reference block in the second reference picture. The block can be predicted by a combination of the first reference block and the second reference block.

[0096] Further, a merge mode technique can be used in the inter-picture prediction to improve coding efficiency.

[0097] According to some embodiments of the disclosure, predictions, such as inter-picture predictions and intra-picture predictions are performed in the unit of blocks. For example, according to the HEVC standard, a picture in a sequence of video pictures is partitioned into coding tree units (CTU) for compression, the CTUs in a picture have the same size, such as 64x64 pixels, 32x32 pixels, or 16x16 pixels. In general, a CTU includes three coding tree blocks (CTBs), which are one luma CTB and two chroma CTBs. Each CTU can be recursively quadtree split into one or multiple coding units (CUs). For example, a CTU of 64x64 pixels can be split into one CU of 64x64 pixels, or 4 CUs of 32x32 pixels, or 16 CUs of 16x16 pixels. In an example, each CU is analyzed to determine a prediction type for the CU, such as an inter prediction type or an intra prediction type. The CU is split into one or more prediction units (PUs) depending on the temporal and/or spatial predictability. Generally, each PU includes a luma prediction block (PB), and two chroma PBs. In an embodiment, a prediction operation in coding (encoding/decoding) is performed in the unit of a prediction block. Using a luma prediction block as an example of a prediction block, the prediction block includes a matrix of values (e.g., luma values) for pixels, such as 8x8 pixels, 16x16 pixels, 8x16 pixels, 16x8 pixels, and the like.

[0098] FIG. 6 shows a diagram of a video encoder (603) according to another embodiment of the disclosure. The video encoder (603) is configured to receive a processing block (e.g., a prediction block) of sample values within a current video picture in a sequence of video pictures, and encode the processing block into a coded picture that is part of a coded video sequence. In an example, the video encoder (603) is used in the place of the video encoder (303) in the FIG. 3 example.

[0099] In an HEVC example, the video encoder (603) receives a matrix of sample values for a processing block, such as a prediction block of 8x8 samples, and the like. The

video encoder (603) determines whether the processing block is best coded using intra mode, inter mode, or bi-prediction mode using, for example, rate-distortion optimization. When the processing block is to be coded in intra mode, the video encoder (603) may use an intra prediction technique to encode the processing block into the coded picture; and when the processing block is to be coded in inter mode or bi-prediction mode, the video encoder (603) may use an inter prediction or bi-prediction technique, respectively, to encode the processing block into the coded picture. In certain video coding technologies, merge mode can be an inter picture prediction submode where the motion vector is derived from one or more motion vector predictors without the benefit of a coded motion vector component outside the predictors. In certain other video coding technologies, a motion vector component applicable to the subject block may be present. In an example, the video encoder (603) includes other components, such as a mode decision module (not shown) to determine the mode of the processing blocks.

[0100] In the FIG. 6 example, the video encoder (603) includes the inter encoder (630), an intra encoder (622), a residue calculator (623), a switch (626), a residue encoder (624), a general controller (621), and an entropy encoder (625) coupled together as shown in FIG. 6.

[0101] The inter encoder (630) is configured to receive the samples of the current block (e.g., a processing block), compare the block to one or more reference blocks in reference pictures (e.g., blocks in previous pictures and later pictures), generate inter prediction information (e.g., description of redundant information according to inter encoding technique, motion vectors, merge mode information), and calculate inter prediction results (e.g., predicted block) based on the inter prediction information using any suitable technique. In some examples, the reference pictures are decoded reference pictures that are decoded based on the encoded video information.

[0102] The intra encoder (622) is configured to receive the samples of the current block (e.g., a processing block), in some cases compare the block to blocks already coded in the same picture, generate quantized coefficients after transform, and in some cases also intra prediction information (e.g., an intra prediction direction information according to one or more intra encoding techniques). In an example, the intra encoder (622) also calculates intra prediction results (e.g., predicted block) based on the intra prediction information and reference blocks in the same picture.

[0103] The general controller (621) is configured to determine general control data and control other components of the video encoder (603) based on the general control data.

In an example, the general controller (621) determines the mode of the block, and provides a control signal to the switch (626) based on the mode. For example, when the mode is the intra mode, the general controller (621) controls the switch (626) to select the intra mode result for use by the residue calculator (623), and controls the entropy encoder (625) to select the intra prediction information and include the intra prediction information in the bitstream; and when the mode is the inter mode, the general controller (621) controls the switch (626) to select the inter prediction result for use by the residue calculator (623), and controls the entropy encoder (625) to select the inter prediction information and include the inter prediction information in the bitstream.

[0104] The residue calculator (623) is configured to calculate a difference (residue data) between the received block and prediction results selected from the intra encoder (622) or the inter encoder (630). The residue encoder (624) is configured to operate based on the residue data to encode the residue data to generate the transform coefficients. In an example, the residue encoder (624) is configured to convert the residue data from a spatial domain to a frequency domain, and generate the transform coefficients. The transform coefficients are then subject to quantization processing to obtain quantized transform coefficients. In various embodiments, the video encoder (603) also includes a residue decoder (628). The residue decoder (628) is configured to perform inverse-transform, and generate the decoded residue data. The decoded residue data can be suitably used by the intra encoder (622) and the inter encoder (630). For example, the inter encoder (630) can generate decoded blocks based on the decoded residue data and inter prediction information, and the intra encoder (622) can generate decoded blocks based on the decoded residue data and the intra prediction information. The decoded blocks are suitably processed to generate decoded pictures and the decoded pictures can be buffered in a memory circuit (not shown) and used as reference pictures in some examples.

[0105] The entropy encoder (625) is configured to format the bitstream to include the encoded block. The entropy encoder (625) is configured to include various information according to a suitable standard, such as the HEVC standard. In an example, the entropy encoder (625) is configured to include the general control data, the selected prediction information (e.g., intra prediction information or inter prediction information), the residue information, and other suitable information in the bitstream. Note that, according to the disclosed subject matter, when coding a block in the merge submode of either inter mode or bi-prediction mode, there is no residue information.

[0106] FIG. 7 shows a diagram of a video decoder (710) according to another embodiment of the disclosure. The video decoder (710) is configured to receive coded pictures that are part of a coded video sequence, and decode the coded pictures to generate reconstructed pictures. In an example, the video decoder (710) is used in the place of the video decoder (310) in the FIG. 3 example.

[0107] In the FIG. 7 example, the video decoder (710) includes an entropy decoder (771), an inter decoder (780), a residue decoder (773), a reconstruction module (774), and an intra decoder (772) coupled together as shown in FIG. 7.

[0108] The entropy decoder (771) can be configured to reconstruct, from the coded picture, certain symbols that represent the syntax elements of which the coded picture is made up. Such symbols can include, for example, the mode in which a block is coded (such as, for example, intra mode, inter mode, bi-predicted mode, the latter two in merge submode or another submode), prediction information (such as, for example, intra prediction information or inter prediction information) that can identify certain sample or metadata that is used for prediction by the intra decoder (772) or the inter decoder (780), respectively, residual information in the form of, for example, quantized transform coefficients, and the like. In an example, when the prediction mode is inter or bi-predicted mode, the inter prediction information is provided to the inter decoder (780); and when the prediction type is the intra prediction type, the intra prediction information is provided to the intra decoder (772). The residual information can be subject to inverse quantization and is provided to the residue decoder (773).

[0109] The inter decoder (780) is configured to receive the inter prediction information, and generate inter prediction results based on the inter prediction information.

[0110] The intra decoder (772) is configured to receive the intra prediction information, and generate prediction results based on the intra prediction information.

[0111] The residue decoder (773) is configured to perform inverse quantization to extract de-quantized transform coefficients, and process the de-quantized transform coefficients to convert the residual from the frequency domain to the spatial domain. The residue decoder (773) may also require certain control information (to include the Quantizer Parameter (QP)), and that information may be provided by the entropy decoder (771) (data path not depicted as this may be low volume control information only).

[0112] The reconstruction module (774) is configured to combine, in the spatial domain, the residual as output by the residue decoder (773) and the prediction results (as output by the inter or intra prediction modules as the case may be) to form a reconstructed

block, that may be part of the reconstructed picture, which in turn may be part of the reconstructed video. It is noted that other suitable operations, such as a deblocking operation and the like, can be performed to improve the visual quality.

[0113] It is noted that the video encoders (303), (503), and (603), and the video decoders (310), (410), and (710) can be implemented using any suitable technique. In an embodiment, the video encoders (303), (503), and (603), and the video decoders (310), (410), and (710) can be implemented using one or more integrated circuits. In another embodiment, the video encoders (303), (503), and (503), and the video decoders (310), (410), and (710) can be implemented using one or more processors that execute software instructions.

[0114] According to some embodiments, a CTU is split into CUs by using a quad tree binary tree (QTBT) structure denoted as a coding tree to adapt to various local characteristics of individual blocks included in the CUs. The decision whether to code a picture area using inter-picture (temporal) or intra-picture (spatial) prediction may be performed at the CU level. Each CU may be further split into one, two or four PUs according to a PU splitting type. In some embodiments, 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 CU may be partitioned into TUs according to another quad tree structure similar to the quad tree structure used for the coding tree for the CTU. In some other embodiments, a PU contains only one TU that has the same shape as the PU.

[0115] The coding tree for the CTU may include multiple partition types including CU, PU, and TU. In some embodiments, a CU or a TU is only a square shape, while a PU may be square or rectangular shape for an inter predicted block. In other embodiments, rectangular shaped CUs, PUs, and TUs are permitted. At a picture boundary, an implicit quad tree split may be applied so that a block will keep quad tree splitting until the size of the split block fits the picture boundary. According to some embodiments, an implicit split means that a split flag is not signaled but implied instead. For example, implicit QT means only a QT split is allowed for a pictureboundary block. As such, the split flag is not signaled at the picture boundary. As an another example, when only a BT split is allowed at the picture boundary, the implicit split is the binary split. In some embodiments, when both QT and BT are allowed at the picture boundary, there is no implicit split, and the split method is explicitly signaled.

[0116] According to some embodiments, the QTBT structure does not include multiple partition types (e.g., QTBT does not include the separation of the CU, PU and TU),

and supports more flexibility for CU partition shapes. For example, in the QTBT block structure, a CU may have either a square or rectangular shape. FIG 8A illustrates an example CTU (800) that is partitioned by the QTBT structure. For example, the CTU (800) is partitioned into four equal sized sub-CUs (A), (B), (C), and (D). FIG. 8B illustrates a corresponding coding tree that illustrates branches corresponding to sub-CUs (A), (B), (C), and (D). The solid lines indicate quad tree splitting, and the dotted lines indicate binary tree splitting. The binary tree structure may include two splitting types: (i) symmetric horizontal splitting and (ii) symmetric vertical splitting. In each splitting (i.e., non-leaf) node of the binary tree, one flag may be signalled to indicate which splitting type (e.g., horizontal or vertical) is used, where 0 indicates horizontal splitting and 1 indicates vertical splitting or vice versa. For the quad tree splitting, the splitting type is not indicated since quad tree splitting splits a block both horizontally and vertically to produce 4 sub-blocks with an equal size.

[0117] As illustrated in FIGs. 8A and 8B, the sub-CU (A) is first partitioned into two sub-blocks by a vertical split, where the left sub-block is partitioned again by another vertical split. The sub-CU (B) is further partitioned by a horizontal split. The sub-CU (C) is further partitioned by another quad split partition. The upper left sub-block of sub-CU (C) is partitioned by a vertical split, and subsequently partitioned by a horizontal split. Furthermore, the lower right sub-block of sub-CU (C) is partitioned by a horizontal split. The upper right and lower left sub-blocks of sub-CU (C) are not further partitioned. The sub-CU (D) is not partitioned further and thus, does not include any additional leaf nodes in the coding tree below the "D" branch.

[0118] The binary tree leaf nodes may be referred to as CUs, where the binary splitting may be used for prediction and transform processing without any further partitioning, which means that the CU, PU, and TU have the same block size in the QTBT coding block structure. A CU may include coding blocks (CBs) of different colour components. For example, one CU may contain one luma CB and two chroma CBs in the case of P and B slices of a 4:2:0 chroma format, and sometimes contain a CB of a single component (e.g., one CU contains only one luma CB or just two chroma CBs in the case of Intra-pictures or I slices). In some embodiments, in intra-pictures or I-slices, the TU width or height is constrained to not exceed a given limit (e.g., 64 for luma and 32 for chroma). If the CB width or height is larger than the limit, then the TU is further split until the TU's size does not exceed the limit.

[0119] According to some embodiments, the QTBT partitioning scheme includes the following parameters:

CTU size: the root node size of a quad tree

MinQTSize: the minimum allowed quad tree leaf node size

MaxBTSize: the maximum allowed binary tree root node size

MaxBTDepth: the maximum allowed binary tree depth

MinBTSize: the minimum allowed binary tree leaf node size

[0120] In one example of the QTBT partitioning structure, the CTU size is set as 128×128 luma samples with two corresponding 64×64 blocks of chroma samples, the *MinQTSize* is set as 16×16, the *MaxBTSize* is set as 64×64, the *MinBTSize* (for both width and height) is set as 4×4, and the *MaxBTDepth* is set as 4. The QTBT partitioning structure is applied to the CTU first to generate quad tree leaf nodes. The quad tree leaf nodes may have a size from 16×16 (i.e., the *MinQTSize*) to 128×128 (i.e., the CTU size). If the leaf quad tree node is 128×128, the leaf quad tree node will not be further split by the binary tree since the size exceeds the *MaxBTSize* (i.e., 64×64). Otherwise, the leaf quad tree node may be further partitioned by the binary tree. Therefore, the quad tree leaf node is also the root node for the binary tree and the quad tree leaf has the binary tree depth as 0. When the binary tree depth reaches the *MaxBTDepth* (e.g., 4), no further splitting is performed. When the binary tree node has width equal to the *MinBTSize* (e.g., 4), no further horizontal splitting is performed. Similarly, when the binary tree node has a height equal to *MinBTSize*, no further vertical splitting is performed. The leaf nodes of the binary tree are further processed by prediction and transform processing without any further partitioning. In some embodiments, the maximum CTU size is 256×256 luma samples.

[0121] The QTBT partition structure may further support the ability for the luma and chroma components to each have separate QTBT structures. For example, for P and B slices, the luma and chroma CTBs in one CTU may share the same QTBT structure. However, for I slices, the luma CTB is partitioned into CUs by a QTBT structure, and the chroma CTBs are partitioned into chroma CUs by another QTBT structure. Therefore, in this example, a CU in an I slice contains a coding block of the luma component or coding blocks of two chroma components, and a CU in a P or B slice contains coding blocks of all three colour components.

[0122] In some embodiments, inter prediction for small blocks is restricted to reduce the memory access requirements of motion compensation, such that bi-prediction is not

supported for 4×8 and 8×4 blocks, and inter prediction is not supported for 4×4 blocks. In other embodiments, the QTBT partition scheme does not include these restrictions.

[0123] According to some embodiments, a Multi-type-tree (MTT) structure includes (i) quad tree splitting, (ii) binary tree splitting, and (iii) horizontal and vertical center-side ternary trees. FIG. 9A illustrates an embodiment of a vertical center-side ternary tree and FIG. 9B illustrates an example of a horizontal center-side ternary tree. Compared to the QTBT structure, MTT can be a more flexible tree structure since additional structures are permitted.

[0124] Ternary tree partitioning includes significantly advantageous features such as providing a complement to quad tree and binary tree partitioning where ternary tree partitioning is able to capture objects which are located in a block center, whereas quad tree and binary tree split along the block center. As another advantage of ternary tree partitioning, the width and height of the partitions of the proposed ternary trees are a power of 2 so that no additional transforms are needed. A two-level tree provides the advantage of complexity reduction. As an example, the complexity of traversing a tree is T^D, where T denotes the number of split types, and D is the depth of tree.

[0125] There are different YUV formats, or chroma formats, which are shown in FIGs. 10A-10D. Each chroma format may define a different down-sampling grid of different color components.

[0126] The color of video samples may be represented in different color formats (e.g., YCbCr or RGB). In the RGB format, the three components (i.e., R, G, and B) have strong correlations, which results in statistical redundancy among the three color components. A color representation of video samples may be converted into a different color space using linear transforms. Converting a RGB color space to a YUV color space may be performed as follows:

Eq. (1): Y = ((66 * R + 129 * G + 25 * B + 128) >> 8) + 16

Eq. (2): U = ((-38 * R - 74 * G + 112 * B + 128) >> 8) + 128

Eq. (3): V = ((112 * R - 94 * G - 18 * B + 128) >> 8) + 128

[0127] Converting a RGB color space to a YUV color space may be performed as follows:

Eq. (4): Y = round(0.256788 * R + 0.504129 * G + 0.097906 * B) + 16

Eq. (5): U = round(-0.148223 * R - 0.290993 * G + 0.439216 * B) + 128

Eq. (6): V = round(0.439216 * R - 0.367788 * G - 0.071427 * B) + 128

[0128] For efficient coding of RGB video content, an in-loop Adaptive Colour Transform (ACT) was develop, where ACT is operated in the residue domain. A CU-level flag may be signaled to indicate the usage of ACT. The forward ACT color transform (e.g., transform performed at encoder) may be performed as follows:

Eq. (7):

$$\begin{bmatrix} Y \\ C_g \\ C_o \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ -1 & 2 & -1 \\ 2 & 0 & -2 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

[0129] The backward ACT color transform (e.g., inverse transform performed at decoder) may be performed as follows:

Eq. (8):

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 \\ 1 & 1 & 0 \\ 1 & -1 & -1 \end{bmatrix} \times \begin{bmatrix} Y \\ C_g \\ C_o \end{bmatrix}$$

[0130] FIG. 11 illustrates an example encoder 1100 for performing a color space transform. In FIG.11, prediction is performed prior to the color space transform being performed. For example, inter prediction or intra prediction is performed on a current block to produce a residual signal. The residual signal is provided a forward color space transform unit (1102) to perform a forward transform such as the transform in Eq. (7). The output of the forward color space transform is provided to a cross component production (CCP) unit (1102). The output of the (CCP) unit (1102) is provided to a transform (T) unit (1106) to perform a transform such as a type of discrete cosine transform (DCT). The output of the transform unit (1106) is provided to a quantizer (Q) (1108) to produce coefficients. The coefficients are provided to an entropy coder unit (1110) to provide a bitstream. The entropy coder unit (1110) may receive a mode/mv signal to select a particular mode of operation of the entropy coder.

[0131] The encoder (1100) may also include the components to convert the bitstream to a residual signal. For example, the bitstream produced by the entropy coder (1110) may be provided to an inverse quantizer (IQ) unit (1112). The output of the inverse quantizer unit (IQ) may be provided to an inverse transform (IT) unit (1114). The output of the inverse transform (IT) unit (1114) may be provided to an inverse CCP unit (1116). The output of the inverse CCP unit (1116) may be provided to an inverse color space transform (1118) where an inverse color transform such as the transform illustrated in Eq. (8) may be performed.

[0132] FIG. 12 illustrates an example decoder (1200) for converting a bitstream to a residual signal. The bitstream illustrated in FIG. 12 may be the bitstream produced by entropy coder (1110)(FIG. 11). The bitstream may be provided to an entropy decoder unit (1200). The output of an entropy decoder unit (1202) may be provided to an inverse an inverse quantizer (IQ) unit (1202). The output of the inverse quantizer unit (IQ) may be provided to an inverse transform (IT) unit (1204). The output of the inverse transform (IT) unit (1204) may be provided to an inverse CCP unit (1206). The output of the inverse CCP unit (1206) may be provided to an inverse color space transform (1208) where an inverse color transform such as the transform illustrated in Eq. (8) may be performed to produce a residual signal. Intra prediction or inter prediction may be performed on the residual signal to decode a current block. The units disclosed in FIGs. 11 and 12 may be implemented in software by a processor or by circuitry such as a specialized integrated circuit designed to perform the function of each unit.

[0133] For the chroma component of an intra PU, the encoder may select the best chroma prediction modes among 8 modes including Planar, DC, Horizontal, Vertical, a direct copy of the intra prediction mode (DM) from the luma component, Left and Top Cross-component Linear Mode (LT_CCLM), Left Cross-component Linear Mode (L_CCLM), and Top Cross-component Linear Mode (T_CCLM). LT_CCLM, L_CCLM, and T_CCLM can be categorized into the group of Cross-component Linear Mode (CCLM). The difference between these 3 modes is that different regions of neighboring samples may be used for deriving the parameters α and β. For LT_CCLM, both the left and top neighboring samples may be used to derive the parameters α and β. For L_CCLM, in some examples, only the left neighboring samples are used to derive the parameters α and β. For T_CCLM, in some examples, only the top neighboring samples are used to derive the parameters α and β.

[0134] Cross-Component Linear Model (CCLM) prediction modes may be used to reduce a cross-component redundancy, in which the chroma samples are predicted based on the reconstructed luma samples of the same CU by using an example linear model as follows:

Eq. (9): $pred_C(i,j) = \alpha \cdot rec_L'(i,j) + \beta,$

where $pred_C(i,j)$ represents the predicted chroma samples in a CU and $rec_L(i,j)$ represents the downsampled reconstructed luma samples of the same CU. Parameters α and β may be derived by a straight line equation, which may also be referred to as a max-min method. Since this computation process may be performed as part of the decoding process, not just as an encoder search operation, no syntax may be used to convey the α and β values.

[0135] For chroma 4:2:0 format, CCLM prediction may apply a six-tap interpolation filter to get the down-sampled luma sample corresponding to a chroma sample as shown in FIG. 13. Based on a formula, a down-sampled luma sample Rec'L[x, y] is calculated from reconstructed luma samples.

[0136] The down-sampled luma samples may be used to find the maximum and minimum sample points. The 2 points (pair of Luma and Chroma) (A, B) may be the minimum and maximum values inside the set of neighboring Luma samples as depicted in FIG 13.

[0137] The linear model parameters α and β may be obtained according to the following equations:

Eq. (10): $\alpha = \frac{y_B - y_A}{x_B - x_A}$

Eq. (11): $\beta = y_A - \alpha x_A$

[0138] A division operation is advantageously avoided by using a multiplication and a shift operation. One Look-up Table (LUT) may be used to store the pre-calculated values, and the absolute difference values between maximum and minimum luma samples may be used to specify the entry index of the LUT. The size of the LUT may be 512.

[0139] FIGs. 14A and 14B illustrate example locations of the samples used for the derivation of α and β in LT_CCLM. In T_CCLM mode, in some examples, only the above neighboring samples (including 2 * W samples) are used to calculate the linear model coefficients. FIGs. 15A and 15B illustrate example locations of the samples used for the derivation of α and β in T_CCLM.

[0140] In L_CCLM mode, in some examples, only left neighboring samples (including 2 * H samples) are used to calculate the linear model coefficients. FIGs. 16A and 16B illustrate example locations of the samples used for the derivation of α and β in L_CCLM.

[0141] The CCLM prediction mode may also include prediction between the two chroma components (i.e., the Cr component is predicted from the Cb component). Instead of using the reconstructed sample signal, the CCLM Cb-to-Cr prediction may be applied in the residual domain. The CCLM Cb-to-Cr prediction may be implemented by adding a weighted reconstructed Cb residual to the original Cr intra prediction to form the final Cr prediction:

Eq. (12): $pred_{Cr}^{*}(i,j) = pred_{Cr}(i,j) + \alpha \cdot resi_{Cb}'(i,j)$

[0142] The CCLM luma-to-chroma prediction mode may be added as one additional chroma intra prediction mode. At the encoder side, one more rate distortion (RD) cost check

for the chroma components is added for selecting the chroma intra prediction mode. When intra prediction modes other than the CCLM luma-to-chroma prediction mode is used for the chroma components of a CU, CCLM Cb-to-Cr prediction is used for Cr component prediction.

[0143] Multiple Model CCLM (MMLM) is another extension of CCLM, where there can be more than one model (e.g., two or more models). In MMLM, neighboring luma samples and neighboring chroma samples of the current block may be classified into two groups, where each group may be used as a training set to derive a linear model (i.e., a particular α and β are derived for a particular group). Furthermore, the samples of the current luma block may also be classified based on the same rule for the classification of neighboring luma samples.

[0144] FIG. 17 shows an example of classifying the neighbouring samples into two groups. The threshold illustrated in FIG. 17 may be calculated as the average value of the neighbouring reconstructed luma samples. A neighbouring sample with Rec'L[x,y] <= Threshold is classified into group 1; while a neighbouring sample with Rec'L[x,y] > Threshold is classified into group 2.

Eq. (13):
$$\begin{cases} Pred_C[x,y] = \alpha_1 \times Rec'_L[x,y] + \beta_1 & if\ Rec'_L[x,y] \le Threshold \\ Pred_C[x,y] = \alpha_2 \times Rec'_L[x,y] + \beta_2 & if\ Rec'_L[x,y] > Threshold \end{cases}$$

[0145] To enable in-loop color transform in VVC for efficient coding of input video with RGB format, interactions between color transform and several coding tools in VVC needs to be handled, such as Cross-component linear Model and Dualtree partitioning. Embodiments of the present disclosure provide the significantly advantageous features of handling color transform with coding tools in VVC.

[0146] The embodiments of the present disclosure may be used separately or combined in any order. Further, each of the methods, encoder, and decoder according to the embodiments of the present disclosure may be implemented by processing circuitry (e.g., one or more processors or one or more integrated circuits). In one example, the one or more processors execute a program that is stored in a non-transitory computer-readable medium. According to embodiments of the present disclosure, the term block may be interpreted as a prediction block, a coding block, or a coding unit (i.e., CU). According to embodiments of the present disclosure, the term luma component may refer to any color component that is coded as the first component in coding order (e.g., (R)ed or (G)reen color component). Furthermore, according to embodiments of the present disclosure, the term chroma

component may refer to any color component that is not coded as the first component in coding order.

[0147] According to some embodiments, a color transform such as ACT is applied before a prediction process is performed at the encoder, and after the reconstruction process is performed at the decoder. At the encoder, the ACT may be performed before prediction (e.g., inter prediction, intra prediction), and the reference samples and input original samples may both be mapped to a different color space if ACT is applied for a current CU. For pixel reconstruction at the decoder, if ACT is applied to a block under reconstruction, the reference samples may be mapped to the alternative color space before being used for prediction, and reconstructed samples may then be mapped back to the original color space.

[0148] FIG. 18 illustrates an embodiment of the encoder and decoder processes using ACT. The units disclosed in FIG. 18 may be implemented in software by a processor or by circuitry such as a specialized integrated circuit designed to perform the function of each unit disclosed in FIG. 18. At the encoder, ACT units (1800) and (1804) perform an ACT transform on both a reference signal and input signal, respectively. The ACT transform performed at the encoder by ACT units (1800) and (1804) may be the ACT transform disclosed in Eq. (7). The output of ACT (1800) is provided to a prediction (P) unit (1802). Furthermore, the reference signal is provided to a prediction (P) unit (1806). The prediction (P) units (1802) and (1806) may perform inter prediction or intra prediction. A transform (T) unit 1808 receives one of (i) a difference between the output of the prediction (P) unit (1802) and the output of the ACT unit (1804) and (ii) a difference between an output of the prediction (P) unit (1806) and the input signal. The transform (T) unit (1808) may perform a transform operation such as a discrete cosine transform (DCT). The output of the transform (T) unit 1808 is provided to a quantizer unit (Q) (1810) to perform a quantization operation to produce a set of coefficients.

[0149] At the decoder, an inverse quantizer (IQ) unit (1812) receives coefficients to perform an inverse quantization process. The output of the inverse quantizer (IQ) unit (1812) is provided to an inverse transform (IT) unit (1814) to perform an inverse transform. An ACT unit (1820) receives a sum of an output of a prediction (P) unit (1818) and the output of the inverse transform (IT) (1814) unit. An ACT unit (1816) receives an output of the prediction (P) unit (1818). The ACT units (1816) and (1820) may perform an inverse color transform such as the inverse color transform disclosed in Eq. (8). The prediction (P) units (1818) and (1822) may perform inter prediction or intra prediction. The reconstructed

reference signal is provided by the output of the ACT unit (1816), and the reconstructed original signal is provided by the output of the ACT unit (1820).

[0150] According to some embodiments, in the ACT process, 2nd and 3rd color components are further offset by a constant c after and before the color transform for forward and backward transform, respectively. Eq. (14) illustrates the modified forward transform, and Eq. (15) illustrates the modified backward (i.e., inverse) transform.

$$\text{Eq. (14):} \begin{bmatrix} Y \\ C_g \\ C_o \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ -1 & 2 & -1 \\ 2 & 0 & -2 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ c \\ c \end{bmatrix}$$

$$\text{Eq. (15):} \begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 \\ 1 & 1 & 0 \\ 1 & -1 & -1 \end{bmatrix} \times \begin{bmatrix} Y \\ C_g - c \\ C_o - c \end{bmatrix}$$

[0151] In some embodiments, the constant c is derived as 1<<(bitDepth-1), where bitDepth refers to the bit-depth of an input sample.

[0152] According to some embodiments, a color transform is only applied when different color components are coded using a same transform unit partitioning tree. In one embodiment, when DualTree is applied on an intra slice, the color transform is applied for an inter slice only.

[0153] According to some embodiments, when a color transform is applied, the CCLM mode is not applied or signaled since the generation of residual samples from one component depends on the reconstruction of another component. In another embodiment, when the CCLM mode is used, the color transform is not applied or signaled. In one embodiment, when the color transform is applied on intra residual samples, the CCLM mode is not applied or signaled since the generation of residual samples from one component depends on the reconstruction of another component. In one embodiment, when the color transform is applied on a residual sample and CCLM mode is used, the color transform is not applied or signaled.

[0154] According to some embodiments, a color transform is signaled for each CTU which is the largest coding unit (CU).

[0155] In some embodiments, a color transform is signaled and applied only for intra coded blocks, or only for inter coded blocks. In some embodiments, when a color transform is applied, the DualTree is not applied (i.e., different color components share the same transform unit partitioning).

[0156] FIG. 19 illustrates an embodiment of a process performed by an encoder such as encoder (603). The process may start at step (S1900), where it is determined if a color

transform condition is satisfied. For example, the color transform condition may be a flag that indicates whether a color transform is enabled for the current block. As another example, the color transform condition may be a flag that indicates that a color transform is enabled for each block in a CTU. If the color transform condition is satisfied, the process proceeds to step (S1902) where a color transform is performed on the current block. For example, the color transform may be the ACT operation illustrated in Eq. (7).

[0157] The process proceeds from step (S1902) to step (S1904) to perform prediction on the color transformed current block. The prediction may be inter prediction or intra prediction. If at step (S1900) the color transform condition is not satisfied, the process from step (S1900) to step (S1904). The process proceeds from step (S1904) to step (S1906) to perform a transform process, such as DCT, on the predicted current block. The process proceeds to step (S1908) to perform a quantization process on the transformed current block. The output of the quantization process may be a set of coefficients that are included in a bitstream that is transmitted to a decoder. The process illustrated in FIG. 19 may terminate after step (S1908) is performed.

[0158] FIG. 20 illustrates an embodiment of a process performed by a decoder such as decoder (710). The process may start at step (S2000) where a coded video bitstream is received. This bitstream may include the coefficients that are generated by step (S1908)(FIG. 19). The process proceeds to step (S2002) where inverse quantization is performed on a set of coefficients corresponding to a current block. The process proceeds to step (S2004) where an inverse transform is performed on the output of the inverse quantization for the current block. The process proceeds to step (S2006) where prediction, such as inter prediction or intra prediction, is performed on the output of the inverse transform corresponding to the current block. At step (S2008), it is determined whether a color transform condition is satisfied for the current block. For example, the color transform condition may be a flag that indicates whether a color transform is enabled for the current block. As another example, the color transform condition may be a flag that indicates that a color transform is enabled for each block in a CTU. If the color transform condition is satisfied, the process proceeds to step (S2010) where an inverse color transform is performed on the current block. For example, the inverse color transform may be the ACT operation illustrated in Eq. (8). If the color transform condition is not satisfied, the process illustrated in FIG. 20 is completed. The process illustrated in FIG. 20 may also be completed after step (S2010) is performed.

[0159] The techniques described above, can be implemented as computer software using computer-readable instructions and physically stored in one or more computer-readable

media. For example, FIG. 21 shows a computer system (2100) suitable for implementing certain embodiments of the disclosed subject matter.

[0160] The computer software can be coded using any suitable machine code or computer language, that may be subject to assembly, compilation, linking, or like mechanisms to create code comprising instructions that can be executed directly, or through interpretation, micro-code execution, and the like, by one or more computer central processing units (CPUs), Graphics Processing Units (GPUs), and the like.

[0161] The instructions can be executed on various types of computers or components thereof, including, for example, personal computers, tablet computers, servers, smartphones, gaming devices, internet of things devices, and the like.

[0162] The components shown in FIG. 21 for computer system (2100) are exemplary in nature and are not intended to suggest any limitation as to the scope of use or functionality of the computer software implementing embodiments of the present disclosure. Neither should the configuration of components be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary embodiment of a computer system (2100).

[0163] Computer system (2100) may include certain human interface input devices. Such a human interface input device may be responsive to input by one or more human users through, for example, tactile input (such as: keystrokes, swipes, data glove movements), audio input (such as: voice, clapping), visual input (such as: gestures), olfactory input (not depicted). The human interface devices can also be used to capture certain media not necessarily directly related to conscious input by a human, such as audio (such as: speech, music, ambient sound), images (such as: scanned images, photographic images obtain from a still image camera), video (such as two-dimensional video, three-dimensional video including stereoscopic video).

[0164] Input human interface devices may include one or more of (only one of each depicted): keyboard (2101), mouse (2102), trackpad (2103), touch screen (2110), data-glove (not shown), joystick (2105), microphone (2106), scanner (2107), camera (2108).

[0165] Computer system (2100) may also include certain human interface output devices. Such human interface output devices may be stimulating the senses of one or more human users through, for example, tactile output, sound, light, and smell/taste. Such human interface output devices may include tactile output devices (for example tactile feedback by the touch-screen (2110), data-glove (not shown), or joystick (2105), but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as:

speakers (2109), headphones (not depicted)), visual output devices (such as screens (2110) to include CRT screens, LCD screens, plasma screens, OLED screens, each with or without touch-screen input capability, each with or without tactile feedback capability—some of which may be capable to output two dimensional visual output or more than three dimensional output through means such as stereographic output; virtual-reality glasses (not depicted), holographic displays and smoke tanks (not depicted)), and printers (not depicted).

[0166] Computer system (2100) can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (2120) with CD/DVD or the like media (2121), thumb-drive (2122), removable hard drive or solid state drive (2123), legacy magnetic media such as tape and floppy disc (not depicted), specialized ROM/ASIC/PLD based devices such as security dongles (not depicted), and the like.

[0167] Those skilled in the art should also understand that term "computer readable media" as used in connection with the presently disclosed subject matter does not encompass transmission media, carrier waves, or other transitory signals.

[0168] Computer system (2100) can also include an interface to one or more communication networks. Networks can for example be wireless, wireline, optical. Networks can further be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of networks include local area networks such as Ethernet, wireless LANs, cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Certain networks commonly require external network interface adapters that attached to certain general purpose data ports or peripheral buses (2149) (such as, for example USB ports of the computer system (2100)); others are commonly integrated into the core of the computer system (2100) by attachment to a system bus as described below (for example Ethernet interface into a PC computer system or cellular network interface into a smartphone computer system). Using any of these networks, computer system (2100) can communicate with other entities. Such communication can be uni-directional, receive only (for example, broadcast TV), uni-directional send-only (for example CANbus to certain CANbus devices), or bi-directional, for example to other computer systems using local or wide area digital networks. Certain protocols and protocol stacks can be used on each of those networks and network interfaces as described above.

[0169] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (2140) of the computer system (2100).

[0170] The core (2140) can include one or more Central Processing Units (CPU) (2141), Graphics Processing Units (GPU) (2142), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (2143), hardware accelerators for certain tasks (2144), and so forth. These devices, along with Read-only memory (ROM) (2145), Random-access memory (2146), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (2147), may be connected through a system bus (2148). In some computer systems, the system bus (2148) can be accessible in the form of one or more physical plugs to enable extensions by additional CPUs, GPU, and the like. The peripheral devices can be attached either directly to the core's system bus (2148), or through a peripheral bus (2149). Architectures for a peripheral bus include PCI, USB, and the like.

[0171] CPUs (2141), GPUs (2142), FPGAs (2143), and accelerators (2144) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (2145) or RAM (2146). Transitional data can also be stored in RAM (2146), whereas permanent data can be stored for example, in the internal mass storage (2147). Fast storage and retrieve to any of the memory devices can be enabled through the use of cache memory, that can be closely associated with one or more CPU (2141), GPU (2142), mass storage (2147), ROM (2145), RAM (2146), and the like.

[0172] The computer readable media can have computer code thereon for performing various computer-implemented operations. The media and computer code can be those specially designed and constructed for the purposes of the present disclosure, or they can be of the kind well known and available to those having skill in the computer software arts.

[0173] As an example and not by way of limitation, the computer system having architecture (2100), and specifically the core (2140) can provide functionality as a result of processor(s) (including CPUs, GPUs, FPGA, accelerators, and the like) executing software embodied in one or more tangible, computer-readable media. Such computer-readable media can be media associated with user-accessible mass storage as introduced above, as well as certain storage of the core (2140) that are of non-transitory nature, such as core-internal mass storage (2147) or ROM (2145). The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core (2140). A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core (2140) and specifically the processors therein (including CPU, GPU, FPGA, and the like) to execute particular processes or particular parts of particular processes described herein, including defining data structures stored in RAM (2146) and modifying such data structures according to the processes defined by the software.

In addition or as an alternative, the computer system can provide functionality as a result of logic hardwired or otherwise embodied in a circuit (for example: accelerator (2144)), which can operate in place of or together with software to execute particular processes or particular parts of particular processes described herein. Reference to software can encompass logic, and vice versa, where appropriate. Reference to a computer-readable media can encompass a circuit (such as an integrated circuit (IC)) storing software for execution, a circuit embodying logic for execution, or both, where appropriate. The present disclosure encompasses any suitable combination of hardware and software.

Appendix A: Acronyms

JEM: joint exploration model

VVC: versatile video coding

BMS: benchmark set

MV: Motion Vector

HEVC: High Efficiency Video Coding

SEI: Supplementary Enhancement Information

VUI: Video Usability Information

GOPs: Groups of Pictures

TUs: Transform Units,

PUs: Prediction Units

CTUs: Coding Tree Units

CTBs: Coding Tree Blocks

PBs: Prediction Blocks

HRD: Hypothetical Reference Decoder

SNR: Signal Noise Ratio

CPUs: Central Processing Units

GPUs: Graphics Processing Units

CRT: Cathode Ray Tube

LCD: Liquid-Crystal Display

OLED: Organic Light-Emitting Diode

CD: Compact Disc

DVD: Digital Video Disc

ROM: Read-Only Memory

RAM: Random Access Memory

ASIC: Application-Specific Integrated Circuit

PLD: Programmable Logic Device

LAN: Local Area Network

GSM: Global System for Mobile communications

LTE: Long-Term Evolution

CANBus: Controller Area Network Bus

USB: Universal Serial Bus

PCI: Peripheral Component Interconnect

FPGA: Field Programmable Gate Areas

SSD: solid-state drive

IC: Integrated Circuit

CU: Coding Unit

[0174] While this disclosure has described several exemplary embodiments, there are alterations, permutations, and various substitute equivalents, which fall within the scope of the disclosure. It will thus be appreciated that those skilled in the art will be able to devise numerous systems and methods which, although not explicitly shown or described herein, embody the principles of the disclosure and are thus within the spirit and scope thereof.

[0175] (1) A method of video decoding performed in a video decoder, the method includes receiving a coded video bitstream including a current picture; performing an inverse quantization on a current block included in the current picture; performing, after performing the inverse quantization, an inverse transform on the current block; performing a prediction process on the current block after performing the inverse transform; after performing the prediction process on the current block, determining whether a predetermined condition is satisfied; and in response to determining that the predetermined condition is met, performing an inverse color transform on the current block.

[0176] (2) The method according to feature (1), in which the inverse color transform is an inverse adaptive color transform (ACT), and performing the inverse color transform converts the reconstructed current block from a color-space transform to a RGB format.

[0177] (3) The method according to feature (1) or (2), in which performing the inverse color transform includes subtracting a constant to one or more color components of the inverse color transform.

[0178] (4) The method according to feature (3), in which the constant is derived from performing a left shift operation on a bit-depth of an input sample minus 1.

[0179] (5) The method of any one of features (1) – (4), in which the predetermined condition is satisfied in response to a determination that a color transform is signaled for the current block.

[0180] (6) The method of feature (5), in which the color transform is signaled for each coding-tree unit (CTU) hat has a largest coding unit (CU).

[0181] (7) The method of any one of features (1) – (6), in which the predetermined condition is satisfied in response to a determination that different color components of the inverse color transform are coded using a same transform unit partitioning tree.

[0182] (8) The method of feature (7), in which in response to a determination that DualTree is applied to intra slice, the inverse color transform is applied only to the inter slice.

[0183] (9) The method of any one of features (1) – (8), further including: in response to the determination that the predetermined condition is satisfied, a cross-component linear mode (CCLM) is not applied to chroma units of the current block.

[0184] (10) The method of video decoding according to any one of features (1) – (9), in which the prediction process is one of inter prediction or intra prediction.

[0185] (11) A video decoder for video decoding including processing circuitry configured to: receive a coded video bitstream including a current picture, perform an inverse quantization on a current block included in the current picture, perform, after performance the inverse quantization, an inverse transform on the current block, perform a prediction process on the current block after performance of the inverse transform, after performance of the prediction process on the current block, determine whether a predetermined condition is satisfied, and in response to a determination that the predetermined condition is met, perform an inverse color transform on the current block.

[0186] (12) The video decoder according to feature (11), in which the inverse color transform is an inverse adaptive color transform (ACT), and the performance of the inverse color transform converts the reconstructed current block from a color-space transform to a RGB format.

[0187] (13) The video decoder according to feature (11) or (12), in which the performance of the inverse color transform includes the processing circuitry configured to subtract a constant to one or more color components of the inverse color transform.

[0188] (14) The video decoder according to feature (13), in which the constant is derived from performing a left shift operation on a bit-depth of an input sample minus 1.

[0189] (15) The video decoder of any one of features (11) – (14), in which the predetermined condition is satisfied in response to a determination that a color transform is signaled for the current block.

[0190] (16) The video decoder of feature (15), in which the color transform is signaled for each coding-tree unit (CTU) hat has a largest coding unit (CU).

[0191] (17) The video decoder of any one of features (11) – (16), in which the predetermined condition is satisfied in response to a determination that different color components of the inverse color transform are coded using a same transform unit partitioning tree.

[0192] (18) The video decoder of feature (17), in which in response to a determination that DualTree is applied to intra slice, the inverse color transform is applied only to the inter slice.

[0193] (19) The video decoder of feature (11), in which in response to the determination that the predetermined condition is satisfied, a cross-component linear mode (CCLM) is not applied to chroma units of the current block.

[0194] (20) A non-transitory computer readable medium having instructions stored therein, which when executed by a processor in a video decoder causes the video decoder to execute a method including: receiving a coded video bitstream including a current picture; performing an inverse quantization on a current block included in the current picture; performing, after performing the inverse quantization, an inverse transform on the current block; performing a prediction process on the current block after performing the inverse transform; after performing the prediction process on the current block, determining whether a predetermined condition is satisfied; and in response to determining that the predetermined condition is met, performing an inverse color transform on the current block.

WHAT IS CLAIMED IS:

1. A method of video decoding performed in a video decoder, the method comprising:

receiving a coded video bitstream including a current picture;

performing an inverse quantization on a current block included in the current picture;

performing, after performing the inverse quantization, an inverse transform on the current block;

performing a prediction process on the current block after performing the inverse transform;

after performing the prediction process on the current block, determining whether a predetermined condition is satisfied; and

in response to determining that the predetermined condition is met, performing an inverse color transform on the current block.

2. The method according to claim 1, wherein the inverse color transform is an inverse adaptive color transform (ACT), and performing the inverse color transform converts the reconstructed current block from a color-space transform to a RGB format.

3. The method according to claim 1, wherein performing the inverse color transform includes subtracting a constant to one or more color components of the inverse color transform.

4. The method according to claim 3, wherein the constant is derived from performing a left shift operation on a bit-depth of an input sample minus 1.

5. The method of claim 1, wherein the predetermined condition is satisfied in response to a determination that a color transform is signaled for the current block.

6. The method of claim 5, wherein the color transform is signaled for each coding-tree unit (CTU) hat has a largest coding unit (CU).

7. The method of claim 1, wherein the predetermined condition is satisfied in response to a determination that different color components of the inverse color transform are coded using a same transform unit partitioning tree.

8. The method of claim 7, wherein in response to a determination that DualTree is applied to intra slice, the inverse color transform is applied only to the inter slice.

9. The method of claim 1, further comprising:

in response to the determination that the predetermined condition is satisfied, a cross-component linear mode (CCLM) is not applied to chroma units of the current block.

10. The method of video decoding according to claim 1, wherein the prediction process is one of inter prediction or intra prediction.

11. A video decoder for video decoding, comprising:

processing circuitry configured to:

receive a coded video bitstream including a current picture,

perform an inverse quantization on a current block included in the current picture,

perform, after performance of the inverse quantization, an inverse transform on the current block,

perform a prediction process on the current block after performance of the inverse transform,

after performance of the prediction process on the current block, determine whether a predetermined condition is satisfied, and

in response to a determination that the predetermined condition is satisfied, perform an inverse color transform on the current block.

12. The video decoder according to claim 11, wherein the inverse color transform is an inverse adaptive color transform (ACT), and the performance of the inverse color transform converts the reconstructed current block from a color-space transform to a RGB format.

13. The video decoder according to claim 11, wherein the performance of the inverse color transform includes the processing circuitry configured to subtract a constant to one or more color components of the inverse color transform.

14. The video decoder according to claim 13, wherein the constant is derived from performing a left shift operation on a bit-depth of an input sample minus 1.

15. The video decoder of claim 11, wherein the predetermined condition is satisfied in response to a determination that a color transform is signaled for the current block.

16. The video decoder of claim 15, wherein the color transform is signaled for each coding-tree unit (CTU) hat has a largest coding unit (CU).

17. The video decoder of claim 11, wherein the predetermined condition is satisfied in response to a determination that different color components of the inverse color transform are coded using a same transform unit partitioning tree.

18. The video decoder of claim 17, wherein in response to a determination that DualTree is applied to intra slice, the inverse color transform is applied only to the inter slice.

19. The video decoder of claim 11, wherein in response to the determination that the predetermined condition is satisfied, a cross-component linear mode (CCLM) is not applied to chroma units of the current block.

20. A non-transitory computer readable medium having instructions stored therein, which when executed by a processor in a video decoder causes the video decoder to execute a method comprising:

receiving a coded video bitstream including a current picture;

performing an inverse quantization on a current block included in the current picture;

performing, after performing the inverse quantization, an inverse transform on the current block;

performing a prediction process on the current block after performing the inverse transform;

after performing the prediction process on the current block, determining whether a predetermined condition is satisfied; and

in response to determining that the predetermined condition is met, performing an inverse color transform on the current block.