METHOD AND APPARATUS FOR INTERACTION BETWEEN INTRA PREDICTION MODE AND BLOCK DIFFERENTIAL PULSE-CODE MODULATION MODE

INCORPORATION BY REFERENCE

[0001] This present application claims the benefit of priority of U.S. Patent Application No. 16/862,221, "METHOD AND APPARATUS FOR INTERACTION BETWEEN INTRA PREDICTION MODE AND BLOCK DIFFERENTIAL PULSE-CODE MODULATION MODE" filed on April 29, 2020, which claims the benefit of priority to U.S. Provisional Application No. 62/841,003, "INTERACTION BETWEEN INTRA PREDICTION MODE AND BLOCK DIFFERENTIAL PULSE-CODE MODULATION MODE" filed on April 30, 2019. The entire disclsoures 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.

SUMMARY

[0019] According to an exemplary embodiment, a method of video decoding performed in a video decoder includes determining whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode. The method further includes, in response to determining that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag. The method further includes determining an inter prediction mode value for the second block using the intra prediction mode value associated with the first block. The method further includes reconstructing the second block using the determined intra prediction mode value.

[0020] According to an exemplary embodiment, a video decoder for video decoding includes processing circuitry configured to: determine whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode. In response to a determination that the first block is coded with the BDPCM mode, the processing circuitry is further configured to associate the first block with an intra prediction mode value based on a BDPCM directional flag. The processing circuitry is further configured to determine an inter prediction mode value for the second block using the intra prediction mode value associated with the first block. The processing circuitry is further configured to reconstruct the second block using the determined intra prediction mode value.

[0021] 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 processor to execute a method that includes determining whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode. The method further includes, in response to determining that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag. The method further includes determining an inter prediction mode value for the second block using the intra prediction mode value associated with the first block. The method further includes reconstructing the second block using the determined intra prediction mode value.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] 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:

[0023] FIG. 1A is a schematic illustration of an exemplary subset of intra prediction modes;

[0024] FIG. 1B is an illustration of exemplary intra prediction directions.

[0025] FIG. 1C is an illustration of exemplary intra prediction directions.

[0026] FIG. 2 is a schematic illustration of a simplified block diagram of a communication system (200) in accordance with an embodiment.

[0027] FIG. 3 is a schematic illustration of a simplified block diagram of a communication system (300) in accordance with an embodiment.

[0028] FIG. 4 is a schematic illustration of a simplified block diagram of a decoder in accordance with an embodiment.

[0029] FIG. 5 is a schematic illustration of a simplified block diagram of an encoder in accordance with an embodiment.

[0030] FIG. 6 shows a block diagram of an encoder in accordance with another embodiment.

[0031] FIG. 7 shows a block diagram of a decoder in accordance with another embodiment.

[0032] FIG. 8 is a schematic illustration of a current block and its surrounding neighbors.

[0033] FIG. 9 is an illustration of an embodiment of a process performed by a decoder.

[0034] FIG. 10 is a schematic illustration of a computer system in accordance with an embodiment of the present disclosure.

DETAILED DESCRIPTION OF EMBODIMENTS

[0035] 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.

[0036] 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.

[0037] 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.

[0038] 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.

[0039] 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.

[0040] 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.

[0041] 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.

[0042] 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).

[0043] 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.

[0044] 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).

[0045] 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.

[0046] 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.

[0047] 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).

[0048] 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).

[0049] 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.

[0050] 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.

[0051] 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.

[0052] 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.

[0053] 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.

[0054] 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.

[0055] 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.

[0056] 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).

[0057] 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.

[0058] 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.

[0059] 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.

[0060] 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).

[0061] 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.

[0062] 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.

[0063] 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).

[0064] 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).

[0065] 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.

[0066] 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.

[0067] 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).

[0068] 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:

[0069] 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.

[0070] 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.

[0071] 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.

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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.

[0076] 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.

[0077] Further, a merge mode technique can be used in the inter-picture prediction to improve coding efficiency.

[0078] 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.

[0079] 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.

[0080] 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.

[0081] 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.

[0082] 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.

[0083] 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.

[0084] 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.

[0085] 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.

[0086] 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.

[0087] 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.

[0088] 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.

[0089] 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).

[0090] The inter decoder (780) is configured to receive the inter prediction information, and generate inter prediction results based on the inter prediction information.

[0091] The intra decoder (772) is configured to receive the intra prediction information, and generate prediction results based on the intra prediction information.

[0092] 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).

[0093] 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.

[0094] 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.

[0095] According to some embodiments, a size of Most Probable Mode (MPM) list is set equal to 6 for both an adjacent reference line (e.g., zero reference line) and non-adjacent reference lines (e.g., non-zero reference lines). The positions of neighboring modes used to derive 6 MPM candidates may also be the same for adjacent and non-adjacent reference lines, which is illustrated in FIG. 8. In FIG. 8, the block A and block B denote the above and left neighboring coding unit of a current block 800, and variables candIntraPredModeA and candIntraPredModeB indicate the associated intra prediction modes of block A and B, respectively. Variables candIntraPredModeA and candIntraPredModeB may initially be set equal to INTRA_PLANAR. If block A (or B) is marked as available, candIntraPredModeA (or candIntraPredModeB) may be set equal to the actual intra prediction mode of block A (or B).

[0096] The MPM candidate derivation process may be different for the adjacent and non-adjacent reference lines. For example, for a zero reference line, if the modes for the two neighboring blocks are Planar or DC mode, default modes are used to construct the MPM list, where the first two candidates are the Planar and DC modes, and the remaining four modes are angular modes (e.g., angular default modes). For non-zero reference lines, if the modes of the two neighboring blocks are Planar or DC mode, 6 angular default modes may be used to construct the MPM list. An embodiment of the MPM list derivation process is shown in Appendix 1, where candModeList[x] with x = 0..5 denotes the 6 MPM candidates, IntraLumaRefLineIdx[xCb][yCb] denotes the reference line index of the block to be predicted, and IntraLumaRefLineIdx[xCb][yCb] can be 0,1, or 3. In some examples, a unified intra mode coding approach is implemented where the planar mode is put as the first MPM.

[0097] Block Differential Pulse Code Modulation (BPDCM) is an intra-coding tool that uses a differential pulse code modulation (DPCM) approach at a block level. In some embodiments, a **bdpcm_flag** is transmitted at a CU level whenever it is a luma intra CU having each dimension smaller or equal to 32. This flag indicates whether regular intra coding or DPCM is used. This flag may be encoded using a single CABAC context.

[0098] In some embodiemnts, BDPCM uses a Median Edge Detector of LOCO-I (used in JPEG-LS). For a current pixel X having pixel A as left neighbor, pixel B as top neighbor, and C as top-left neighbor, the prediction P(X) may be determined by:

$$\text{Eq. (1): } P(X) = \begin{cases} \min(A,B) & \text{if } C \geq \max(A,B) \\ \max(A,B) & \text{if } C \leq \min(A,B) \\ A+B-C & \text{otherwise} \end{cases}$$

[0099] The predictor may use unfiltered reference pixels when predicting from the top row and left column of the CU. The predictor then may use reconstructed pixels for the rest of the CU. Pixels may be processed in raster-scan order inside the CU. The prediction error may be quantized in the spatial domain, after rescaling, in a way identical to a Transform Skip quantizer. Each pixel may be reconstructed by adding the dequantized prediction error to the prediction. Thus, the reconstructed pixels may be used to predict the next pixels in raster-scan order. Amplitude and signs of the quantized prediction error may be encoded separately.

[0100] In some embodiments, a **cbf_bdpcm_flag** is coded. If this flag is equal to 0, all amplitudes of the block may be decoded as zero. If this flag is equal to 1, all amplitudes of the

block may be encoded individually in raster-scan order. In order to keep complexity low, in some examples, the amplitude may be limited to at most 31 (inclusive). The amplitude may be encoded using unary binarization, with three contexts for the first bin, then one context for each additional bin until the 12th bin, and one context for all remaining bins. A sign may be encoded in bypass mode for each zero residue.

[0101] In some embodiments, to maintain the coherence of the regular intra mode prediction, the first mode in the MPM list is associated with a Block-DPCM CU (without being transmitted) and is available for MPM generation for subsequent blocks. The deblocking filter may be de-activated on a border between two BDPCM blocks, since neither of the blocks uses the transform stage usually responsible for blocking artifacts. In some embodiments, BDPCM does not use any other step than the ones disclosed herein. For example, BPDCM does not use any transform.

[0102] According to some embodiments, a BDPCM method uses reconstructed samples to predict the rows or columns of a CU line by line. The signalled BDPCM direction may indicate whether vertical or horizontal prediction is used. The reference pixels used may be unfiltered samples. The prediction error may be quantized in the spatial domain. Pixels may be reconstructed by adding the dequantized prediction error to the prediction.

[0103] In some embodiments, as an alternative scheme to BDPCM, quantized residual domain BDPCM may be performed. The signalling and prediction directions used in quantized residual BDPCM may be identical to a BPCM scheme. The intra prediction may be performed on an entire block by sample copying in a prediction direction (horizontal or vertical prediction) similar to intra prediction. The residual may be quantized, and the delta between the quantized residual and the quantized residual's predictor (horizontal or vertical) quantized value may be coded, which can be described in the following disclosed embodiments.

[0104] For a block of size M (rows) × N (cols), let $r_{i,j}$, $0 \le i \le M-1$, $0 \le j \le N-1$ be the prediction residual after performing intra prediction horizontally (copying left neighbor pixel value across the the predicted block line by line) or vertically (copying top neighbor line to each line in the predicted block) using unfiltered samples from above or left block boundary samples. Let $Q(r_{i,j})$, $0 \le i \le M-1$, $0 \le j \le N-1$ denote the quantized version of the residual $r_{i,j}$, where a residual is a difference between an original block and the predicted block values. Then, BDPCM is applied to the quantized residual samples, resulting in a modified M × N array $\tilde{R}$ with elements $\tilde{r}_{i,j}$. In some examples, when vertical BDPCM is signalled:

Eq. (2):

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

[0105] In some examples, for horizontal prediction, similar rules apply, and the residual quantized samples may be obtained by:

Eq. (3):

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

[0106] The residual quantized samples $\tilde{r}_{i,j}$ may be sent to the decoder. On the decoder side, in some examples the above calculations are reversed to produce $Q(r_{i,j}$), $0 \le i \le M-1$, $0 \le j \le N-1$. In some embodiments, for a vertical prediction case:

Eq. (4):

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

[0107] In some embodiments, for a horizontal prediction case:

Eq. (5):

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

[0108] The inverse quantized residuals, $Q^{-1}(Q(r_{i,j}))$, may be added to the intra block prediction values to produce the reconstructed sample values. One advantage to this scheme is that the inverse DPCM may be performed on the fly during coefficient parsing by adding the predictor as the coefficients are parsed or it can be performed after parsing. Therefore, the splitting of 4xN and Nx4 blocks into 2 parallel processed blocks can be eliminated.

[0109] In some embodiments, a BDPCM coded block is associated with an intra prediction mode which is the first MPM (i.e., MPM0). As a result, when deriving the MPM list, if a neighboring block is coded with BDPCM mode, its associated intra prediction mode (i.e., MPM0) is used. In addition, when a chroma block is coded using DM mode and the co-located luma block is coded using a BDPCM mode, the intra prediction mode associated with the co-located luma block (i.e., MPM0) is used as the intra prediction mode of the current chroma block.

[0110] Table 1 (below) illustrates an embodiment of the syntax and semantics of the BDPCM method.

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	**Descriptor**
if(tile_group_type != I || sps_ibc_enabled_flag) {	
if(treeType != DUAL_TREE_CHROMA)	
cu_skip_flag[x0][y0]	ae(v)
if(cu_skip_flag[x0][y0] = = 0 && tile_group_type != I)	
pred_mode_flag	ae(v)
if(((tile_group_type = = I && cu_skip_flag[x0][y0] = =0) || (tile_group_type != I && CuPredMode[x0][y0] != MODE_INTRA)) && sps_ibc_enabled_flag)	
pred_mode_ibc_flag	ae(v)
}	
if(CuPredMode[x0][y0] = = MODE_INTRA) {	
if(pred_mode_flag = = MODE_INTRA && (cIdx == 0) && (cbWidth <= 32) && (CbHeight <= 32)) {	
bdpcm_flag[x0][y0]	ae(v)
if(bdpcm_flag[x0][y0]) {	
bdpcm_dir_flag[x0][y0]	ae(v)
}	
else {	
if(sps_pcm_enabled_flag && cbWidth >= MinIpcmCbSizeY && cbWidth <= MaxIpcmCbSizeY && cbHeight >= MinIpcmCbSizeY && cbHeight <= MaxIpcmCbSizeY)	
pcm_flag[x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned( ))	
pcm_alignment_zero_bit	f(1)
pcm_sample(cbWidth, cbHeight, treeType)	
} else {	
if(treeType = = SINGLE_TREE || treeType = = DUAL_TREE_LUMA) {	
if((y0 % CtbSizeY) > 0)	
intra_luma_ref_idx[x0][y0]	ae(v)
if (intra_luma_ref_idx[x0][y0] = = 0 && (cbWidth <= MaxTbSizeY || cbHeight <= MaxTbSizeY) && (cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))	
intra_subpartitions_mode_flag[x0][y0]	ae(v)
if(intra_subpartitions_mode_flag[x0][y0] = = 1 && cbWidth <= MaxTbSizeY && cbHeight <= MaxTbSizeY)	
intra_subpartitions_split_flag[x0][y0]	ae(v)
if(intra_luma_ref_idx[x0][y0] = = 0 && intra_subpartitions_mode_flag[x0][y0] = = 0)	
intra_luma_mpm_flag[x0][y0]	ae(v)
if(intra_luma_mpm_flag[x0][y0])	
intra_luma_mpm_idx[x0][y0]	ae(v)
Else	
intra_luma_mpm_remainder[x0][y0]	ae(v)

}	
}	
if(treeType = = SINGLE_TREE || treeType = = DUAL_TREE_CHROMA)	
intra_chroma_pred_mode[x0][y0]	ae(v)
}	
} else if(treeType != DUAL_TREE_CHROMA) { /* MODE_INTER or MODE_IBC */	
……	
}	
}	

Table 1

[0111] In some embodiments, the variable **bdpcm_flag**[x0][y0] equal to 1 specifies that a bdpcm_dir_flag is present in the coding unit including the luma coding block at the location (x0, y0). In some embodiments, **bdpcm_dir_flag**[x0][y0] equal to 0 specifies that the prediction direction to be used in a bdpcm block is horizontal, otherwise the prediction direction is vertical.

[0112] As understood by one of ordinary skill in the art, BDPCM contributes significant coding gain on screen video content, which is typically featured by strong edges. However, when BDPCM is jointly used with the MPM or DM modes, a BDPCM coded block is always associated with the Planar mode, which may be harmful for coding gain on screen video content. Embodiments of the present disclosure resolve these disadvantages.

[0113] 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).

[0114] According to some embodiments, when the bdpcm_dir_flag is equal to 0, a horizontal prediction is used for BDPCM residual prediction, and when the bdpcm_dir_flag is equal to 1, the vertical prediction is used for BDPCM residual prediction. However, in other embodiments, the opposite approach also applies when the prediction directions of bdpcm_dir_flag being equal to 0 and 1 are swapped.

[0115] In some embodiments, the horizontal intra prediction mode is represented using HOR_IDX, where in VVC, HOR_IDX corresponds to the intra prediction mode

INTRA_ANGULAR18, and in HEVC, HOR_IDX corresponds to intra prediction mode INTRA_ANGULAR10. In some embodiments, the vertical intra prediction mode is represented using VER_IDX, where in VVC, VER_IDX corresponds to the intra prediction mode INTRA_ANGULAR50, and in HEVC, VER_IDX corresponds to the intra prediction mode INTRa_ANGULAR26.

[0116] According to some embodiments, when deriving the most probable intra prediction modes, if a neighboring block is coded by BDPCM mode, the neighboring block is associated with an intra prediction mode *ipm* which is derived using the value of bdpcm_dir_flag applied for this BDPCM coded neighboring block as follows:

Eq. (6): *ipm* = bdpcm_dir_flag == 0 ? HOR_IDX : VER_IDX,

where HOR_IDX and VER_IDX represents the horizontal and vertical intra prediction mode, respectively, and bdpcm_dir_flag equals to 0 indicates that horizontal prediction is used for BDPCM residual prediction, and bdpcm_dir_flag equals to 1 indicates that vertical prediction is used for BDPCM residual prediction. After a value is assigned to intra prediction mode value *ipm*, this intra prediction mode value is considered as the neighboring block intra prediction mode and used for deriving the most probable intra prediction mode of a current block.

[0117] Appendix 2 illustrates an embodiment of the MPM list derivation process in which the bold portion illustrates how an intra prediction mode for a BDPCM coded block is determined based on the bdpcm_dir_flag. Example input to this process may include: (i) a luma location (xCb , yCb) specifying the top-left sample of the current luma coding block relative to the top left luma sample of the current picture, (ii) a variable cbWidth specifying the width of the current coding block in luma samples, (iii) a variable cbHeight specifying the height of the current coding block in luma samples. In this process of Appendix 2, the luma intra prediction mode IntraPredModeY[xCb][yCb] is derived.

[0118] Table 2 specifies the value for the intra prediction mode IntraPredModeY[xCb][yCb] and the associated names. In Table 2, in some examples, the intra prediction modes INTRA_LT_CCLM, INTRA_L_CCLM and INTRA_T_CCLM, are only applicable to chroma components.

Intra prediction mode	Associated name
0	INTRA_PLANAR
1	INTRA_DC
2..66	INTRA_ANGULAR2..INTRA_ANGULAR66
81..83	INTRA_LT_CCLM, INTRA_L_CCLM, INTRA_T_CCLM

Table 2

[0119] According to some embodiments, when deriving the intra prediction mode of a chroma block while the chroma block's co-located luma block is coded using BDPCM mode, if the chroma block is predicted using DM mode, the intra prediction mode used for performing intra prediction mode of this chroma block is derived as follows,

Eq. (7): *dm* = bdpcm_dir_flag==0 ? HOR_IDX : VER_IDX,

where HOR_IDX and VER_IDX represents the horizontal and vertical intra prediction mode, respectively, and bdpcm_dir_flag equals to 0 indicates that horizontal prediction is used for BDPCM residual prediction, and bdpcm_dir_flag equals to 1 indicates that vertical prediction is used for BDPCM residual prediction. Accordingly, after the value *dm* is assigned, this value is used as the intra prediction mode of the chroma block.

[0120] According to some embodiments, the context used for entropy coding the bdpcm_dir_flag depends on the value of bdpcm_dir_flag of neighboring blocks and/or whether the neighboring block is coded by the horizontal intra prediction mode or vertical intra prediction mode.

[0121] In one embodiment, only the bdpcm_dir_flag and bdpcm_flag values of a neighboring block is used for deriving the context applied for entropy coding the bdpcm_dir_flag of a current block. In one example, two neighboring blocks are used (i.e., left is block A in FIG. 8 and top is block B in FIG. 8), and the context value (ctx) is derived as follows:

Eq. (8): dpcm_left = $dpcm_flag_{left}$? ($bdpcm_dir_flag_{left}$? 1 : 2) : 0

Eq. (9): dpcm_top = $dpcm_flag_{top}$? ($bdpcm_dir_flag_{top}$? 1 : 2) : 0

Eq. (10): ctx = dpcm_left * 3 + dpcm_top,

where $dpcm_flag_{left}$ and $dpcm_flag_{top}$ refer to the dpcm_flag of the left and top neighboring blocks, respectively, and $bdpcm_dir_flag_{left}$ and $bdpcm_dir_flag_{top}$ refer to the bdpcm_dir_flag of the left and top neighboring blocks, respectively. After ctx is assigned, this value may be used as an index for selecting one of a plurality of context models.

[0122] In addition to the previous example, the 9 contexts may be grouped in a pre-defined manner resulting in less contexts being applied. For example, ctx=8 and ctx=7 in the previous example may be merged and only one context may be used for both of these ctx values.

[0123] In some embodiments, two neighboring blocks are used, (i.e., left and top), and the context value (ctx) is derived as follows, where $dpcm_flag_{left}$ and $dpcm_flag_{top}$ refer to the dpcm_flag of the left and top neighboring blocks, respectively, and $bdpcm_dir_flag_{left}$ and $bdpcm_dir_flag_{top}$ refer to the bdpcm_dir_flag of the left and top neighboring blocks, respectively.

Eq. (11): $dpcm_left = dpcm_flag_{left} \; ? \; (bdpcm_dir_flag_{left} \; ? \; 1 : 2) : 0$

Eq. (12): $dpcm_top = dpcm_flag_{top} \; ? \; (bdpcm_dir_flag_{top} \; ? \; 1 : 2) : 0$

Eq. (13): $ctx = (dpcm_left == dpcm_top) \; ? \; dpcm_left : 0.$

[0124] FIG. 9 illustrates an embodiment of a process performed by a decoder such as video decoder (710). The process may start at step (S900) to determine whether a first block associated with a second block is coded in the BDPCM mode. In some examples, the first block may be a block that is a spatial neighbor of a second block located in the same picture as the first block. In other examples, the first block may be a luma block and the second block is a chroma block, where the luma block is co-located with the chroma block.

[0125] If the first block is coded in the BDPCM mode, the process proceeds from step (S900) to step (S902), where the first block is associated with an intra prediction mode value based on a BDPCM directional flag. For example, bdpcm_flag may indicate that a block is coded in the BDPCM mode, and bdpcm_dir_flag may be used to determine whether to use the horizontal direction or vertical direction. The process proceeds to step (S904) to determine an inter prediction mode value for the second block using the intra prediction mode value associated with the first block. For example, based on the BDPCM directional flag, the intra prediction mode value may be one of a horizontal intra prediction mode value and a vertical intra prediction mode value. Furthermore, if the first block is a spatial neighbor of the second block, the intra prediction mode value of the first block may be used for creating a MPM list, where the MPM list is used for deriving the intra prediction mode value of the second block. Additionally, if the second block is a chroma block that is predicted using the DM mode, and the first block is a co-located luma block, the intra prediction mode value of the second block may be determined based on the intra prediction mode value of the first block.

[0126] The process proceeds to step (S906) where the second block is reconstructed using the determined intra prediction mode value of the second block. The process illustrated in FIG. 9 may terminate after step (S906) is completed. Furthermore, returning to step (S900), if the first block is not coded in BDPCM mode, the process illustrated in FIG. 9 may terminate.

[0127] 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. 10 shows a computer system (1000) suitable for implementing certain embodiments of the disclosed subject matter.

[0128] 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.

[0129] 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.

[0130] The components shown in FIG. 10 for computer system (1000) 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 (1000).

[0131] Computer system (1000) 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).

[0132] Input human interface devices may include one or more of (only one of each depicted): keyboard (1001), mouse (1002), trackpad (1003), touch screen (1010), data-glove (not shown), joystick (1005), microphone (1006), scanner (1007), camera (1008).

[0133] Computer system (1000) 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 (1010), data-glove (not shown), or joystick (1005), but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers (1009), headphones (not depicted)), visual output devices (such as screens (1010) 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).

[0134] Computer system (1000) can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (1020) with CD/DVD or the like media (1021), thumb-drive (1022), removable hard drive or solid state drive (1023), 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.

[0135] 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.

[0136] Computer system (1000) 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 (1049) (such as, for example USB ports of the computer system (1000)); others are commonly

integrated into the core of the computer system (1000) 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 (1000) 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.

[0137] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (1040) of the computer system (1000).

[0138] The core (1040) can include one or more Central Processing Units (CPU) (1041), Graphics Processing Units (GPU) (1042), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (1043), hardware accelerators for certain tasks (1044), and so forth. These devices, along with Read-only memory (ROM) (1045), Random-access memory (1046), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (1047), may be connected through a system bus (1048). In some computer systems, the system bus (1048) 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 (1048), or through a peripheral bus (1049). Architectures for a peripheral bus include PCI, USB, and the like.

[0139] CPUs (1041), GPUs (1042), FPGAs (1043), and accelerators (1044) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (1045) or RAM (1046). Transitional data can also be stored in RAM (1046), whereas permanent data can be stored for example, in the internal mass storage (1047). 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 (1041), GPU (1042), mass storage (1047), ROM (1045), RAM (1046), and the like.

[0140] 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.

[0141] As an example and not by way of limitation, the computer system having architecture (1000), and specifically the core (1040) 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 (1040) that are of non-transitory nature, such as core-internal mass storage (1047) or ROM (1045). The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core (1040). A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core (1040) 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 (1046) 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 (1044)), 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.

[0142] 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.

[0143] (1) A method of video decoding performed in a video decoder, the method including determining whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode, in response to determining that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag; determining an inter prediction mode value for the

second block using the intra prediction mode value associated with the first block; and reconstructing the second block using the determined intra prediction mode value.

[0144] (2) The method according to feature (1), in which the BDPCM directional flag is one of (i) a first value associated with a horizontal intra prediction directional mode, and (ii) a second value associated with a vertical intra prediction directional mode.

[0145] (3) The method according to feature (2), in which the total number of intra prediction modes is 67 with the horizontal intra prediction directional mode associated with angular mode 18 and with the vertical intra prediction directional mode associated with angular mode 50.

[0146] (4) The method according to any one of features (1) – (3), in which determining whether the first block is coded with the BDPCM mode is based on a value of a BDPCM flag that indicates the presence of the BDPCM directional flag.

[0147] (5) The method according to any one of features (1) – (4), in which the first block and second block are included in a same picture, and the first block spatially neighbors the second block.

[0148] (6) The method according to feature (5), further including: deriving for the second block, using a most probable mode (MPM) derivation process, a candidate list, the deriving including the determining whether the first block is coded with the BDPCM mode, in which determining the inter prediction mode value for the second block further includes using the derived candidate list.

[0149] (7) The method according to feature (6), in which the candidate list includes a first candidate intra prediction mode value ($Mode_1$) that corresponds to the intra prediction mode of the first block, and a second candidate intra prediction mode value ($Mode_2$) and a third candidate intra prediction mode value ($Mode_3$) that are determined in accordance with a predetermined offset from the first candidate intra prediction mode value and a modulo M operation, in which M is a power of 2.

[0150] (8) The method according to any one of features (1) – (7), in which the second block is a chroma block and the first block is a luma block co-located with the chroma block.

[0151] (9) The method according to feature (8), the method further including determining whether the second block is coded with a direct copy mode (DM); and in response to determining that the second block is coded with the direct copy mode, determining whether the first block is coded with the BDPCM mode.

[0152] (10) A video decoder for video decoding, including processing circuitry configured to: determine whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode, in response to a determination that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag, determine an inter prediction mode value for the second block using the intra prediction mode value associated with the first block, and reconstruct the second block using the determined intra prediction mode value.

[0153] (11) The video decoder according to feature (10), in which the BDPCM directional flag is one of (i) a first value associated with a horizontal intra prediction directional mode, and (ii) a second value associated with a vertical intra prediction directional mode.

[0154] (12) The video decoder according to feature (11), in which the total number of intra prediction modes is 67 with the horizontal intra prediction directional mode associated with angular mode 18 and with the vertical intra prediction directional mode associated with angular mode 50.

[0155] (13) The video decoder according to any one of features (10) – (12), in which the determination whether the first block is coded with the BDPCM mode is based on a value of a BDPCM flag that indicates the presence of the BDPCM directional flag.

[0156] (14) The video decoder according to any one of features (10) – (13), in which the first block and second block are included in a same picture, and the first block spatially neighbors the second block.

[0157] (15) The video decoder according to feature (14), in which the processing circuitry is further configured to: derive for the second block, using a most probable mode (MPM) derivation process, a candidate list, the derivation including the determination of whether the first block is coded with the BDPCM mode, in which the determination of the inter prediction mode value for the second block further includes using the derived candidate list.

[0158] (16) The video decoder according to feature (15), in which the candidate list includes a first candidate intra prediction mode value ($Mode_1$) that corresponds to the intra prediction mode of the first block, and a second candidate intra prediction mode value ($Mode_2$) and a third candidate intra prediction mode value ($Mode_3$) that are determined in accordance with a predetermined offset from the first candidate intra prediction mode value and a modulo M operation, in which M is a power of 2.

[0159] (17) The video decoder according to any one of features (10), the in which the second block is a chroma block and the first block is a luma block co-located with the chroma block.

[0160] (18) The video decoder according to feature (17), in which the processing circuitry is further configured to: determine whether the second block is coded with a direct copy mode (DM), and in response to a determination that the second block is coded with the direct copy mode, determine whether the first block is coded with the BDPCM mode.

[0161] (19) 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 determining whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode, in response to determining that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag; determining an inter prediction mode value for the second block using the intra prediction mode value associated with the first block; and reconstructing the second block using the determined intra prediction mode value.

[0162] (20) The non-transitory computer readable medium according to feature 19, in which the BDPCM directional flag is one of (i) a first value associated with a horizontal intra prediction directional mode, and (ii) a second value associated with a vertical intra prediction directional mode.

APPENDIX 1

- If candIntraPredModeB is equal to candIntraPredModeA and candIntraPredModeA is greater than INTRA_DC, candModeList[x] with x = 0..5 is derived as follows:
  - If IntraLumaRefLineIdx[xCb][yCb] is equal to 0, the following applies:

candModeList[0] = candIntraPredModeA (A1_4)

candModeList[1] = INTRA_PLANAR (A1_5)

candModeList[2] = INTRA_DC (A1_6)

candModeList[3] = 2 + ((candIntraPredModeA + 61) % 64) (A1_7)

candModeList[4] = 2 + ((candIntraPredModeA − 1) % 64) (A1_8)

candModeList[5] = 2 + ((candIntraPredModeA + 60) % 64) (A1_9)

– Otherwise (IntraLumaRefLineIdx[xCb][yCb] is not equal to 0), the following applies:

candModeList[0] = candIntraPredModeA (A1_10)

candModeList[1] = 2 + ((candIntraPredModeA + 61) % 64)(A1_11)

candModeList[2] = 2 + ((candIntraPredModeA − 1) % 64) (A1_12)

candModeList[3] = 2 + ((candIntraPredModeA + 60) % 64)(A1_13)

candModeList[4] = 2 + (candIntraPredModeA % 64) (A1_14)

candModeList[5] = 2 + ((candIntraPredModeA + 59) % 64)(A1_15)

– Otherwise if candIntraPredModeB is not equal to candIntraPredModeA and candIntraPredModeA or candIntraPredModeB is greater than INTRA_DC, the following applies:

– The variables minAB and maxAB are derived as follows:

minAB = candModeList[(candModeList[0] > candModeList[1]) ? 1 : 0] (A1_16)

maxAB = candModeList[(candModeList[0] > candModeList[1]) ? 0 : 1] (A1_17)

– If candIntraPredModeA and candIntraPredModeB are both greater than INTRA_DC, candModeList[x] with x = 0..5 is derived as follows:

candModeList[0] = candIntraPredModeA (A1_18)

candModeList[1] = candIntraPredModeB (A1_19)

– If IntraLumaRefLineIdx[xCb][yCb] is equal to 0, the following applies:

candModeList[2] = INTRA_PLANAR (A1_20)

candModeList[3] = INTRA_DC (A1_21)

– If maxAB − minAB is in the range of 2 to 62, inclusive, the following applies:

candModeList[4] = 2 + ((maxAB + 61) % 64) (A1_22)

candModeList[5] = 2 + ((maxAB − 1) % 64) (A1_23)

– Otherwise, the following applies:

candModeList[4] = 2 + ((maxAB + 60) % 64) (A1_24)

candModeList[5] = 2 + ((maxAB) % 64) (A1_25)

- Otherwise (IntraLumaRefLineIdx[xCb][yCb] is not equal to 0), the following applies:
  - If maxAB − minAB is equal to 1, the following applies:
    candModeList[2] = 2 + ((minAB + 61) % 64) (A1_26)
    candModeList[3] = 2 + ((maxAB − 1) % 64) (A1_27)
    candModeList[4] = 2 + ((minAB + 60) % 64) (A1_28)
    candModeList[5] = 2 + (maxAB % 64) (A1_29)
  - Otherwise if maxAB − minAB is equal to 2, the following applies:
    candModeList[2] = 2 + ((minAB − 1) % 64) (A1_30)
    candModeList[3] = 2 + ((minAB + 61) % 64) (A1_31)
    candModeList[4] = 2 + ((maxAB − 1) % 64) (A1_32)
    candModeList[5] = 2 + ((minAB + 60) % 64) (A1_33)
  - Otherwise if maxAB − minAB is greater than 61, the following applies:
    candModeList[2] = 2 + ((minAB − 1) % 64) (A1_34)
    candModeList[3] = 2 + ((maxAB + 61) % 64) (A1_35)
    candModeList[4] = 2 + (minAB % 64) (A1_36)
    candModeList[5] = 2 + ((maxAB + 60) % 64) (A1_37)
  - Otherwise, the following applies:
    candModeList[2] = 2 + ((minAB + 61) % 64) (A1_38)
    candModeList[3] = 2 + ((minAB − 1) % 64) (A1_39)
    candModeList[4] = 2 + ((maxAB + 61) % 64) (A1_40)
    candModeList[5] = 2 + ((maxAB − 1) % 64) (A1_41)
- Otherwise (candIntraPredModeA or candIntraPredModeB is greater than INTRA_DC), candModeList[x] with x = 0..5 is derived as follows:
  - If IntraLumaRefLineIdx[xCb][yCb] is equal to 0, the following applies:
    candModeList[0] = candIntraPredModeA (A1_42)
    candModeList[1] = candIntraPredModeB (A1_43)
    candModeList[2] = 1 − minAB (A1_44)
    candModeList[3] = 2 + ((maxAB + 61) % 64) (A1_45)
    candModeList[4] = 2 + ((maxAB − 1) % 64) (A1_46)

candModeList[5] = 2 + ((maxAB + 60) % 64) (A1_47)

– Otherwise (IntraLumaRefLineIdx[xCb][yCb] is not equal to 0), the following applies:

candModeList[0] = maxAB (A1_48)

candModeList[1] = 2 + ((maxAB + 61) % 64) (A1_49)

candModeList[2] = 2 + ((maxAB − 1) % 64) (A1_50)

candModeList[3] = 2 + ((maxAB + 60) % 64) (A1_51)

candModeList[4] = 2 + (maxAB % 64) (A1_52)

candModeList[5] = 2 + ((maxAB + 59) % 64) (A1_53)

– Otherwise, the following applies:

– If IntraLumaRefLineIdx[xCb][yCb] is equal to 0, the following applies:

candModeList[0] = candIntraPredModeA (A1_54)

candModeList[1] =

(candModeList[0] == INTRA_PLANAR) ? INTRA_DC : (A1_55)
INTRA_PLANAR

candModeList[2] = INTRA_ANGULAR50 (A1_56)

candModeList[3] = INTRA_ANGULAR18 (A1_57)

candModeList[4] = INTRA_ANGULAR46 (A1_58)

candModeList[5] = INTRA_ANGULAR54 (A1_59)

– Otherwise (IntraLumaRefLineIdx[xCb][yCb] is not equal to 0), the following applies:

candModeList[0] = INTRA_ANGULAR50 (A1_60)

candModeList[1] = INTRA_ANGULAR18 (A1_61)

candModeList[2] = INTRA_ANGULAR2 (A1_62)

candModeList[3] = INTRA_ANGULAR34 (A1_63)

candModeList[4] = INTRA_ANGULAR66 (A1_64)

candModeList[5] = INTRA_ANGULAR26 (A1_65)

APPENDIX 2

IntraPredModeY[xCb][yCb] is derived by the following ordered steps:

1. The neighbouring locations (xNbA, yNbA) and (xNbB, yNbB) are set equal to (xCb − 1, yCb + cbHeight − 1) and (xCb + cbWidth − 1, yCb − 1), respectively.
2. For X being replaced by either A or B, the variables candIntraPredModeX are derived as follows:
   - The availability derivation process for a block is invoked with the location (xCurr, yCurr) set equal to (xCb, yCb) and the neighbouring location (xNbY, yNbY) set equal to (xNbX, yNbX) as inputs, and the output is assigned to availableX.
   - The candidate intra prediction mode candIntraPredModeX is derived as follows:
     - If one or more of the following conditions are true, candIntraPredModeX is set equal to INTRA_PLANAR.
       - The variable availableX is equal to FALSE.
       - CuPredMode[xNbX][yNbX] is not equal to MODE_INTRA and ciip_flag[xNbX][yNbX] is not equal to 1.
       - pcm_flag[xNbX][yNbX] is equal to 1.
       - X is equal to B and yCb − 1 is less than ((yCb >> CtbLog2SizeY) << CtbLog2SizeY).
     - Otherwise, candIntraPredModeX is set equal to IntraPredModeY[xNbX][yNbX].
3. The variables ispDefaultMode1 and ispDefaultMode2 are defined as follows:
   - If IntraSubPartitionsSplitType is equal to ISP_HOR_SPLIT, ispDefaultMode1 is set equal to INTRA_ANGULAR18 and ispDefaultMode2 is set equal to INTRA_ANGULAR5.
   - Otherwise, ispDefaultMode1 is set equal to INTRA_ANGULAR50 and ispDefaultMode2 is set equal to INTRA_ANGULAR63.
4. The candModeList[x] with x = 0..5 is derived as follows:

– If candIntraPredModeB is equal to candIntraPredModeA and candIntraPredModeA is greater than INTRA_DC, candModeList[x] with x = 0..5 is derived as follows:

  – If IntraLumaRefLineIdx[xCb][yCb] is equal to 0 and IntraSubPartitionsSplitType is equal to ISP_NO_SPLIT, the following applies:

    candModeList[0] = candIntraPredModeA (A2_9)

    candModeList[1] = INTRA_PLANAR (A2_10)

    candModeList[2] = INTRA_DC (A2_11)

    candModeList[3] = 2 + ((candIntraPredModeA + 61) % 64) (A2_12)

    candModeList[4] = 2 + ((candIntraPredModeA − 1) % 64) (A2_13)

    candModeList[5] = 2 + ((candIntraPredModeA + 60) % 64) (A2_14)

  – Otherwise (IntraLumaRefLineIdx[xCb][yCb] is not equal to 0 or IntraSubPartitionsSplitType is not equal to ISP_NO_SPLIT), the following applies:

    candModeList[0] = candIntraPredModeA (A2_15)

    candModeList[1] = 2 + ((candIntraPredModeA + 61) % 64) (A2_16)

    candModeList[2] = 2 + ((candIntraPredModeA − 1) % 64) (A2_17)

    – If one of the following conditions is true,

      – IntraSubPartitionsSplitType is equal to ISP_HOR_SPLIT and candIntraPredModeA is less than INTRA_ANGULAR34,

      – IntraSubPartitionsSplitType is equal to ISP_VER_SPLIT and candIntraPredModeA is greater than or equal to INTRA_ANGULAR34,

      – IntraLumaRefLineIdx[xCb][yCb] is not equal to 0,

      the following applies:

      candModeList[3] = 2 + ((candIntraPredModeA + 60) % 64)(A2_18)

candModeList[4] = 2 + (candIntraPredModeA % 64) (A2_19)

candModeList[5] = 2 + ((candIntraPredModeA + 59) % 64)(A2_20)

– Otherwise, the following applies:

candModeList[3] = ispDefaultMode1 (A2_21)

candModeList[4] = ispDefaultMode2 (A2_22)

candModeList[5] = INTRA_PLANAR (A2_23)

– Otherwise if candIntraPredModeB is not equal to candIntraPredModeA and candIntraPredModeA or candIntraPredModeB is greater than INTRA_DC, the following applies:

– The variables minAB and maxAB are derived as follows:

minAB = Min(candIntraPredModeA, candIntraPredModeB) (A2_24)

maxAB = Max(candIntraPredModeA, candIntraPredModeB) (A2_25)

– If candIntraPredModeA and candIntraPredModeB are both greater than INTRA_DC, candModeList[x] with x = 0..5 is derived as follows:

candModeList[0] = candIntraPredModeA (A2_26)

candModeList[1] = candIntraPredModeB (A2_27)

– If IntraLumaRefLineIdx[xCb][yCb] is equal to 0 and IntraSubPartitionsSplitType is equal to ISP_NO_SPLIT, the following applies:

candModeList[2] = INTRA_PLANAR (A2_28)

candModeList[3] = INTRA_DC (A2_29)

– If maxAB − minAB is in the range of 2 to 62, inclusive, the following applies:

candModeList[4] = 2 + ((maxAB + 61) % 64) (A2_30)

candModeList[5] = 2 + ((maxAB − 1) % 64) (A2_31)

– Otherwise, the following applies:

candModeList[4] = 2 + ((maxAB + 60) % 64) (A2_32)

candModeList[5] = 2 + ((maxAB) % 64) (A2_33)

– Otherwise (IntraLumaRefLineIdx[xCb][yCb] is not equal to 0 or IntraSubPartitionsSplitType is not equal to ISP_NO_SPLIT), the following applies:

– When IntraSubPartitionsSplitType is not equal to ISP_NO_SPLIT, and abs(candIntraPredModeB − ispDefaultMode1) is less than abs(candIntraPredModeA − ispDefaultMode1), the following applies:

candModeList[0] = candIntraPredModeB (A2_34)

candModeList[1] = candIntraPredModeA (A2_35)

– If maxAB − minAB is equal to 1, the following applies:

candModeList[2] = 2 + ((minAB + 61) % 64) (A2_36)

candModeList[3] = 2 + ((maxAB − 1) % 64) (A2_37)

candModeList[4] = 2 + ((minAB + 60) % 64) (A2_38)

candModeList[5] = 2 + (maxAB % 64) (A2_39)

– Otherwise if maxAB − minAB is equal to 2, the following applies:

candModeList[2] = 2 + ((minAB − 1) % 64) (A2_40)

candModeList[3] = 2 + ((minAB + 61) % 64) (A2_41)

candModeList[4] = 2 + ((maxAB − 1) % 64) (A2_42)

candModeList[5] = 2 + ((minAB + 60) % 64) (A2_43)

– Otherwise if maxAB − minAB is greater than 61, the following applies:

candModeList[2] = 2 + ((minAB − 1) % 64) (A2_44)

candModeList[3] = 2 + ((maxAB + 61) % 64) (A2_45)

candModeList[4] = 2 + (minAB % 64) (A2_46)

candModeList[5] = 2 + ((maxAB + 60) % 64) (A2_47)

– Otherwise, the following applies:

candModeList[2] = 2 + ((minAB + 61) % 64) (A2_48)

candModeList[3] = 2 + ((minAB − 1) % 64) (A2_49)

candModeList[4] = 2 + ((maxAB + 61) % 64) (A2_50)

candModeList[5] = 2 + ((maxAB − 1) % 64) (A2_51)

– Otherwise (candIntraPredModeA or candIntraPredModeB is greater than INTRA_DC), candModeList[x] with x = 0..5 is derived as follows:

– If IntraLumaRefLineIdx[xCb][yCb] is equal to 0 and IntraSubPartitionsSplitType is equal to ISP_NO_SPLIT, the following applies:

candModeList[0] = candIntraPredModeA (A2_52)

candModeList[1] = candIntraPredModeB (A2_53)

candModeList[2] = 1 − minAB (A2_54)

candModeList[3] = 2 + ((maxAB + 61) % 64) (A2_55)

candModeList[4] = 2 + ((maxAB − 1) % 64) (A2_56)

candModeList[5] = 2 + ((maxAB + 60) % 64) (A2_57)

– Otherwise, if IntraLumaRefLineIdx[xCb][yCb] is not equal to 0, the following applies:

candModeList[0] = maxAB (A2_58)

candModeList[1] = 2 + ((maxAB + 61) % 64) (A2_59)

candModeList[2] = 2 + ((maxAB − 1) % 64) (A2_60)

candModeList[3] = 2 + ((maxAB + 60) % 64) (A2_61)

candModeList[4] = 2 + (maxAB % 64) (A2_62)

candModeList[5] = 2 + ((maxAB + 59) % 64) (A2_63)

– Otherwise (IntraSubPartitionsSplitType is not equal to ISP_NO_SPLIT), the following applies:

candModeList[0] = INTRA_PLANAR (A2_64)

candModeList[1] = maxAB (A2_65)

candModeList[2] = 2 + ((maxAB + 61) % 64) (A2_66)

candModeList[3] = 2 + ((maxAB − 1) % 64) (A2_67)

candModeList[4] = 2 + ((maxAB + 60) % 64) (A2_68)

candModeList[5] = 2 + (maxAB % 64) (A2_69)

– Otherwise, the following applies:

– If IntraLumaRefLineIdx[xCb][yCb] is equal to 0 and IntraSubPartitionsSplitType is equal to ISP_NO_SPLIT, the following applies:

candModeList[0] = candIntraPredModeA (A2_70)

candModeList[1] =
(candModeList[0] == INTRA_PLANAR) ? INTRA_DC : (A2_71)
INTRA_PLANAR

candModeList[2] = INTRA_ANGULAR50 (A2_72)

candModeList[3] = INTRA_ANGULAR18 (A2_73)

candModeList[4] = INTRA_ANGULAR46 (A2_74)

candModeList[5] = INTRA_ANGULAR54 (A2_75)

– Otherwise, if IntraLumaRefLineIdx[xCb][yCb] is not equal to 0, the following applies:

candModeList[0] = INTRA_ANGULAR50 (A2_76)

candModeList[1] = INTRA_ANGULAR18 (A2_77)

candModeList[2] = INTRA_ANGULAR2 (A2_78)

candModeList[3] = INTRA_ANGULAR34 (A2_79)

candModeList[4] = INTRA_ANGULAR66 (A2_80)

candModeList[5] = INTRA_ANGULAR26 (A2_81)

– Otherwise, if IntraSubPartitionsSplitType is equal to ISP_HOR_SPLIT, the following applies:

candModeList[0] = INTRA_PLANAR (A2_82)

candModeList[1] = INTRA_ANGULAR18 (A2_83)

candModeList[2] = INTRA_ANGULAR25 (A2_84)

candModeList[3] = INTRA_ANGULAR10 (A2_85)

candModeList[4] = INTRA_ANGULAR65 (A2_86)

candModeList[5] = INTRA_ANGULAR50 (A2_87)

– Otherwise, if IntraSubPartitionsSplitType is equal to ISP_VER_SPLIT, the following applies:

candModeList[0] = INTRA_PLANAR (A2_88)

candModeList[1] = INTRA_ANGULAR50 (A2_89)

candModeList[2] = INTRA_ANGULAR43 (A2_90)

candModeList[3] = INTRA_ANGULAR60 (A2_91)

candModeList[4] = INTRA_ANGULAR3 (A2_92)

candModeList[5] = INTRA_ANGULAR18 (A2_93)

5. IntraPredModeY[xCb][yCb] is derived by applying the following procedure:

   - If bdpcm_flag[xCb][yCb] is equal to 1, the IntraPredModeY[xCb][yCb] is set equal to **bdpcm_dir_flag[xCb][yCb] = = 0 ? INTRA_ANGULAR18: INTRA_ANGULAR50.**
   - Otherwise if intra_luma_mpm_flag[xCb][yCb] is equal to 1, the IntraPredModeY[xCb][yCb] is set equal to candModeList[intra_luma_mpm_idx[xCb][yCb]].
   - Otherwise, IntraPredModeY[xCb][yCb] is derived by applying the following ordered steps:

     1. When candModeList[i] is greater than candModeList[j] for i = 0..4 and for each i, j = (i + 1)..5, both values are swapped as follows:

        (candModeList[i], candModeList[j]) = Swap(candModeList[i], candModeList[j]) (A2_94)

     2. IntraPredModeY[xCb][yCb] is derived by the following ordered steps:

        i. IntraPredModeY[xCb][yCb] is set equal to intra_luma_mpm_remainder[xCb][yCb].

        ii. For i equal to 0 to 5, inclusive, when IntraPredModeY[xCb][yCb] is greater than or equal to candModeList[i], the value of IntraPredModeY[xCb][yCb] is incremented by one.

The variable IntraPredModeY[x][y] with x = xCb..xCb + cbWidth − 1 and y = yCb..yCb + cbHeight − 1 is set to be equal to IntraPredModeY[xCb][yCb].

WHAT IS CLAIMED IS:

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

determining whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode,

in response to determining that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag;

determining an inter prediction mode value for the second block using the intra prediction mode value associated with the first block; and

reconstructing the second block using the determined intra prediction mode value.

2. The method according to claim 1, wherein the BDPCM directional flag is one of (i) a first value associated with a horizontal intra prediction directional mode, and (ii) a second value associated with a vertical intra prediction directional mode.

3. The method according to claim 2, wherein the total number of intra prediction modes is 67 with the horizontal intra prediction directional mode associated with angular mode 18 and with the vertical intra prediction directional mode associated with angular mode 50.

4. The method according to claim 1, wherein determining whether the first block is coded with the BDPCM mode is based on a value of a BDPCM flag that indicates the presence of the BDPCM directional flag.

5. The method according to claim 1, wherein the first block and second block are included in a same picture, and the first block spatially neighbors the second block.

6. The method according to claim 5,further comprising:

deriving for the second block, using a most probable mode (MPM) derivation process, a candidate list, the deriving including the determining whether the first block is coded with the BDPCM mode, wherein

determining the inter prediction mode value for the second block further includes using the derived candidate list.

7. The method according to claim 6, wherein the candidate list includes

a first candidate intra prediction mode value ($Mode_1$) that corresponds to the intra prediction mode of the first block, and

a second candidate intra prediction mode value ($Mode_2$) and a third candidate intra prediction mode value ($Mode_3$) that are determined in accordance with a predetermined offset from the first candidate intra prediction mode value and a modulo M operation, in which M is a power of 2.

8. The method according to claim 1, wherein the second block is a chroma block and the first block is a luma block co-located with the chroma block.

9. The method according to claim 8, further comprising:

determining whether the second block is coded with a direct copy mode (DM); and

in response to determining that the second block is coded with the direct copy mode, determining whether the first block is coded with the BDPCM mode.

10. A video decoder for video decoding, comprising:

processing circuitry configured to:

determine whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode,

in response to a determination that the first block is coded with the BDPCM mode, associate the first block with an intra prediction mode value based on a BDPCM directional flag,

determine an inter prediction mode value for the second block using the intra prediction mode value associated with the first block, and

reconstruct the second block using the determined intra prediction mode value.

11. The video decoder according to claim 10, wherein the BDPCM directional flag is one of (i) a first value associated with a horizontal intra prediction directional mode, and (ii) a second value associated with a vertical intra prediction directional mode.

12. The video decoder according to claim 11, wherein the total number of intra prediction modes is 67 with the horizontal intra prediction directional mode associated with angular mode 18 and with the vertical intra prediction directional mode associated with angular mode 50.

13. The video decoder according to claim 10, wherein the determination whether the first block is coded with the BDPCM mode is based on a value of a BDPCM flag that indicates the presence of the BDPCM directional flag.

14. The video decoder according to claim 10, wherein the first block and second block are included in a same picture, and the first block spatially neighbors the second block.

15. The video decoder according to claim 14, wherein the processing circuitry is further configured to:

derive for the second block, using a most probable mode (MPM) derivation process, a candidate list, the derivation including the determination of whether the first block is coded with the BDPCM mode, wherein

the determination of the inter prediction mode value for the second block further includes using the derived candidate list.

16. The video decoder according to claim 15, wherein the candidate list includes

a first candidate intra prediction mode value ($Mode_1$) that corresponds to the intra prediction mode of the first block, and

a second candidate intra prediction mode value ($Mode_2$) and a third candidate intra prediction mode value ($Mode_3$) that are determined in accordance with a predetermined offset from the first candidate intra prediction mode value and a modulo M operation, in which M is a power of 2.

17. The video decoder according to claim 10, the wherein the second block is a chroma block and the first block is a luma block co-located with the chroma block.

18. The video decoder according to claim 17, wherein the processing circuitry is further configured to:

determine whether the second block is coded with a direct copy mode (DM), and

in response to a determination that the second block is coded with the direct copy mode, determine whether the first block is coded with the BDPCM mode.

19. 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:

determining whether a first block associated with a second block is coded with a block differential pulse code modulation (BDPCM) mode,

in response to determining that the first block is coded with the BDPCM mode, associating the first block with an intra prediction mode value based on a BDPCM directional flag;

determining an inter prediction mode value for the second block using the intra prediction mode value associated with the first block; and

reconstructing the second block using the determined intra prediction mode value.

20. The non-transitory computer readable medium according to claim 19, wherein the BDPCM directional flag is one of (i) a first value associated with a horizontal intra prediction

directional mode, and (ii) a second value associated with a vertical intra prediction directional mode.