METHOD AND APPARATUS FOR SIGNALING PREDICTOR CANDIDATE LIST SIZE

INCORPORATION BY REFERENCE

[0001] This present disclosure claims the benefit of priority to U.S. Patent Application No. 16/844,339, "METHOD AND APPARATUS FOR SIGNALING PREDICTOR CANDIDATE LIST SIZE," filed April 9, 2020, which claims the benefit of priority to U.S. Provisional Application No. 62/846,375, "PREDICTOR CANDIDATE LIST SIZE SIGNALING FOR INTRA PICTURE BLOCK COMPENSATION" filed on May 10, 2019 and U.S. Provisional Application No. 62/845,712, "MERGE LIST SIZE SIGNALING FOR INTRA PICTURE BLOCK COMPENSATION" filed on May 9, 2019. The entire disclosures of the prior applications are hereby incorporated by reference in their entirety.

TECHNICAL FIELD

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

BACKGROUND

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

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

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

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

[0006] Motion compensation can be a lossy compression technique and can relate to techniques where a block of sample data from a previously reconstructed picture or part thereof (reference picture), after being spatially shifted in a direction indicated by a motion vector (MV henceforth), is used for the prediction of a newly reconstructed picture or picture part. In some cases, the reference picture can be the same as the picture currently under reconstruction. MVs can have two dimensions X and Y, or three dimensions, the third being an indication of the reference picture in use (the latter, indirectly, can be a time dimension).

[0007] In some video compression techniques, an MV applicable to a certain area of sample data can be predicted from other MVs, for example from those related to another area of sample data spatially adjacent to the area under reconstruction, and preceding that MV in decoding order. Doing so can substantially reduce the amount of data required for coding the MV, thereby removing redundancy and increasing compression. MV prediction can work effectively, for example, because when coding an input video signal derived from a camera (known as natural video) there is a statistical likelihood that areas larger than the area to which a single MV is applicable move in a similar direction and, therefore, can in some cases be predicted using a similar motion vector derived from MVs of neighboring area. That results in the MV found for a given area to be similar or the same as the MV predicted from the surrounding MVs, and that in turn can be represented, after entropy coding, in a smaller number of bits than what would be used if coding the MV directly. In some cases, MV prediction can be an example of lossless compression of a signal (namely: the MVs) derived from the original signal (namely: the sample stream). In other cases, MV prediction itself can be lossy, for example because of rounding errors when calculating a predictor from several surrounding MVs.

[0008] Various MV prediction mechanisms are described in H.265/HEVC (ITU-T Rec. H.265, "High Efficiency Video Coding", December 2016). Out of the many MV prediction

mechanisms that H.265 offers, described here is a technique henceforth referred to as "spatial merge".

[0009] Referring to FIG. 1, a current block (101) comprises samples that have been found by the encoder during the motion search process to be predictable from a previous block of the same size that has been spatially shifted. Instead of coding that MV directly, the MV can be derived from metadata associated with one or more reference pictures, for example from the most recent (in decoding order) reference picture, using the MV associated with either one of five surrounding samples, denoted A0, A1, and B0, B1, B2 (102 through 106, respectively). In H.265, the MV prediction can use predictors from the same reference picture that the neighboring block is using. The order of forming a candidate list may be A0 → B0 → B1 → A1 → B2

SUMMARY

[0010] According to an exemplary embodiment, a method of video decoding includes receiving a coded video bitstream including a current picture. The method further includes determining whether a current block included in the current picture is coded in intra block copy (IBC) mode. The method further includes, in response to the current block being coded in the IBC mode, determining a number of IBC predictor candidates associated with the current block. The method further includes constructing an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates. The method further includes selecting a block vector predictor from the IBC predictor candidate list. The method further includes decoding a block vector associated with the current block using the block vector predictor. The method further includes decoding the current block in accordance with the block vector.

[0011] According to an exemplary embodiment, a video decoder for performing video decoding includes processing circuitry configured to receive a coded video bitstream including a current picture. The processing circuitry is further configured to determine whether a current block included in the current picture is coded in intra block copy (IBC) mode, in response to the current block being coded in the IBC mode. The processing circuitry is further configured to determine a number of IBC predictor candidates associated with the current block. The processing circuitry is further configured to construct an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates. The processing circuitry is further configured to select a block vector predictor from the IBC predictor candidate list. The processing circuitry is further configured to decode a block vector associated with the current

block using the block vector predictor. The processing circuitry is further configured to decode the current block in accordance with the block vector.

[0012] 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 receiving a coded video bitstream including a current picture. The method further includes determining whether a current block included in the current picture is coded in intra block copy (IBC) mode. The method further includes, in response to the current block being coded in the IBC mode, determining a number of IBC predictor candidates associated with the current block. The method further includes constructing an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates. The method further includes selecting a block vector predictor from the IBC predictor candidate list. The method further includes decoding a block vector associated with the current block using the block vector predictor. The method further includes decoding the current block in accordance with the block vector.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] 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:

[0014] FIG. 1 is a schematic illustration of a current block and its surrounding spatial merge candidates in one example.

[0015] FIG. 2 is a schematic illustration of a simplified block diagram of a communication system (200) in accordance with an embodiment.

[0016] FIG. 3 is a schematic illustration of a simplified block diagram of a communication system (300) in accordance with an embodiment.

[0017] FIG. 4 is a schematic illustration of a simplified block diagram of a decoder in accordance with an embodiment.

[0018] FIG. 5 is a schematic illustration of a simplified block diagram of an encoder in accordance with an embodiment.

[0019] FIG. 6 shows a block diagram of an encoder in accordance with another embodiment.

[0020] FIG. 7 shows a block diagram of a decoder in accordance with another embodiment.

[0021] FIG. 8 is a schematic illustration of intra picture block compensation in accordance with an embodiment.

[0022] FIGs. 9A-9D is a schematic illustration of intra picture block compensation with one coding tree unit (CTU) size search range in accordance with an embodiment.

[0023] FIGS. 10A-10D is a schematic illustration of how a buffer is updated in accordance with embodiments.

[0024] FIG. 11A is an illustration of a decoding flow chart for a history-based MV prediction (HMVP) buffer.

[0025] FIG. 11B is a schematic illustration of updating a HMVP buffer.

[0026] FIG. 12 is an illustration of an example decoding process in accordance with an embodiment.

[0027] FIG. 13 is a schematic illustration of a computer system in accordance with an embodiment of the present disclosure.

DETAILED DESCRIPTION OF EMBODIMENTS

[0028] 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.

[0029] 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.

[0030] 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.

[0031] 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.

[0032] 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.

[0033] 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.

[0034] 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.

[0035] 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).

[0036] 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.

[0037] 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).

[0038] 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.

[0039] 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.

[0040] 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).

[0041] 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).

[0042] 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.

[0043] 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.

[0044] 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.

[0045] 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.

[0046] 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.

[0047] 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.

[0048] 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.

[0049] 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).

[0050] 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.

[0051] 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.

[0052] 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.

[0053] 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).

[0054] 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.

[0055] 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.

[0056] 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).

[0057] 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).

[0058] 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.

[0059] 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.

[0060] 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).

[0061] 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:

[0062] 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.

[0063] 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.

[0064] 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.

[0065] 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.

[0066] 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.

[0067] 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.

[0068] 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.

[0069] 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.

[0070] Further, a merge mode technique can be used in the inter-picture prediction to improve coding efficiency.

[0071] 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.

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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.

[0076] 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.

[0077] 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.

[0078] 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.

[0079] 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.

[0080] 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.

[0081] 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.

[0082] 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).

[0083] The inter decoder (780) is configured to receive the inter prediction information, and generate inter prediction results based on the inter prediction information.

[0084] The intra decoder (772) is configured to receive the intra prediction information, and generate prediction results based on the intra prediction information.

[0085] 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).

[0086] 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.

[0087] 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.

[0088] Block based compensation from a different picture may be referred to as motion compensation. Block compensation may also be done from a previously reconstructed area within the same picture, which may be referred to as intra picture block compensation, intra block copy (IBC), or current picture referencing (CPR). For example, a displacement vector that indicates an offset between a current block and the reference block is referred to as a block vector. According to some embodiments, a block vector points to a reference block that is already reconstructed and available for reference. Also, for parallel processing consideration, a reference area that is beyond a tile/slice boundary or wavefront ladder-shaped boundary may also be excluded from being referenced by the block vector. Due to these constraints, a block vector may be different from a motion vector in motion compensation, where the motion vector can be at any value (positive or negative, at either x or y direction).

[0089] The coding of a block vector may be either explicit or implicit. In an explicit mode, which is sometimes referred to as (Advanced Motion Vector Prediction) AMVP mode in inter coding, the difference between a block vector and its predictor is signaled. In the implicit mode, the block vector is recovered from the block vector's predictor, in a similar way as a motion vector in merge mode. The resolution of a block vector, in some embodiments, is restricted to integer positions. In other embodiments, the resolution of a block vector may be allowed to point to fractional positions.

[0090] The use of intra block copy at the block level may be signaled using a block level flag, referred to as an IBC flag. In one embodiment, the IBC flag is signaled when a current block is not coded in merge mode. The IBC flag may also be signaled by a reference index approach, which is performed by treating the current decoded picture as a reference picture. In HEVC Screen Content Coding (SCC), such a reference picture is put in the last position of the list. This special reference picture may also be managed together with other temporal reference pictures in the DPB. IBC may also include variations such as flipped IBC (e.g., the reference

block is flipped horizontally or vertically before used to predict current block), or line based (IBC) (e.g., each compensation unit inside an MxN coding block is an Mx1 or 1xN line).

[0091] FIG. 8 illustrates an embodiment of intra picture block compensation (e.g., intra block copy mode). In FIG. 8, a current picture 800 includes a set of block regions that have already been coded/decoded (i.e., gray colored squares) and a set of block regions that have yet to be coded/decoded (i.e., white colored squares). A block 802 of one of the block regions that have yet to be coded/decoded may be associated with a block vector 804 that points to another block 806 that has previously been coded/decoded. Accordingly, any motion information associated with the block 806 may be used for the coding/decoding of block 802.

[0092] In some embodiments, the search range of the CPR mode is constrained to be within the current CTU. The effective memory requirement to store reference samples for CPR mode is 1 CTU size of samples. Taking into account the existing reference sample memory to store reconstructed samples in a current 64x64 region, 3 more 64x64 sized reference sample memory are required. Embodiments of the present disclosure extend the effective search range of the CPR mode to some part of the left CTU while the total memory requirement for storing reference pixels are kept unchanged (1 CTU size, 4 64x64 reference sample memory in total).

[0093] In FIG. 9A, the upper left region of CTU 900 is the current region being decoded. When the upper left region of CTU 900 is decoded, the entry [1] of the reference sample memory is overwritten with the samples from this region, as illustrated in FIG 10A (e.g., over-written memory location(s) has diagonal cross-hatching). In FIG. 9B, the upper right region of CTU 900 is the next current region being decoded. When the upper right region of CTU 900 is decoded, the entry [2] of the reference sample memory is overwritten with the samples from this region, as illustrated in FIG 10B. In FIG. 9C, the lower left region of CTU 900 is the next current region being decoded. When the lower left region of CTU 900 is decoded, the entry [3] of the reference sample memory is overwritten with the samples from this region, as illustrated in FIG 10C. In FIG. 9D, the lower right region of CTU 900 is the next current region being decoded. When the lower right region of CTU 900 is decoded, the entry [3] of the reference sample memory is overwritten with the samples from this region, as illustrated in FIG 10D.

[0094] In some embodiments, the bitstream conformance conditions that a valid block vector (mvL, in 1/16 – pel resolution) should follow the conditions specified below. In some embodiments, the luma motion vector mvL obeys the following constraints A1. A2, B1, C1, and C2.

[0095] In a first constraint (**A1**), when the derivation process for block availability (e.g., neighbouring blocks availability checking process) is invoked with the current luma location (xCurr, yCurr) set equal to (xCb, yCb) and the neighbouring luma location (xCb + (mvL[0] >> 4) + cbWidth − 1, yCb + (mvL[1] >> 4) + cbHeight − 1) as inputs, the output shall be equal to TRUE.

[0096] In a second constraint (**A2**), when the derivation process for block availability (e.g., neighbouring blocks availability checking process) is invoked with the current luma location (xCurr, yCurr) set equal to (xCb, yCb) and the neighbouring luma location (xCb + (mvL[0] >> 4) + cbWidth − 1, yCb + (mvL[1] >> 4) + cbHeight − 1) as inputs, the the output shall be equal to TRUE.

[0097] In a third constraint (**B1**), one or both of the following conditions are true:

(i) The value of (mvL[0] >> 4) + cbWidth is less than or equal to 0.

(ii) The value of (mvL[1] >> 4) + cbHeight is less than or equal to 0.

[0098] In a fourth constraint (**C1**), the following conditions are true:

(i) (yCb + (mvL[1] >> 4)) >> CtbLog2SizeY = yCb >> CtbLog2SizeY

(ii) (yCb + (mvL[1] >> 4) + cbHeight − 1) >> CtbLog2SizeY = yCb >> CtbLog2SizeY

(iii) (xCb + (mvL[0] >> 4)) >> CtbLog2SizeY >= (xCb >> CtbLog2SizeY) − 1

(iv) (xCb + (mvL[0] >> 4) + cbWidth − 1) >> CtbLog2SizeY <= (xCb >> CtbLog2SizeY)

[0099] In a fifth constraint (**C2**), when (xCb + (mvL[0] >> 4)) >> CtbLog2SizeY is equal to (xCb >> CtbLog2SizeY) − 1, the derivation process for block availability (e.g., neighbouring blocks availability checking process) is invoked with the current luma location(xCurr, yCurr) set equal to (xCb, yCb) and the neighbouring luma location (((xCb + (mvL[0] >> 4) + CtbSizeY) >> (CtbLog2SizeY − 1)) << (CtbLog2SizeY − 1), ((yCb + (mvL[1] >> 4)) >> (CtbLog2SizeY − 1)) << (CtbLog2SizeY − 1)) as inputs, and the output shall be equal to FALSE.

[0100] In the above equations, xCb and yCb are the x and y coordinates of the current block, respectively. The variables cbHeight and cbWidth are the height and width of the current block, respectively. The variable CtbLog2sizeY refers to the CTU size in the log2 domain. For example, CtbLog2sizeY = 7 means that the CTU size is 128 x 128. The variables mvL0[0] and mvL0[1] refer to the x and y components of block vector mvL0, respectively. If an output is

FALSE, the samples for a reference block are determined to be available (e.g., neighboring block is available for intra block copy usage). If an output is TRUE, the samples for a reference block are determined to be not available.

[0101] According to some embodiments, a history-based MVP (HMVP) method includes a HMVP candidate that is defined as the motion information of a previously coded block. A table with multiple HMVP candidates is maintained during the encoding/decoding process. The table is emptied when a new slice is encountered. Whenever there is an inter-coded non-affine block, the associated motion information is added to the last entry of the table as a new HMVP candidate. The coding flow of the HMVP method is depicted in FIG. 11A.

[0102] The table size S is set to be 6, which indicates up to 6 HMVP candidates may be added to the table. When inserting a new motion candidate into the table, a constrained FIFO rule is utilized such that a redundancy check is first applied to determine whether an identical HMVP is in the table. If found, the identical HMVP is removed from the table and all the HMVP candidates afterwards are moved forward, i.e., with indices reduced by 1. FIG. 11B shows an example of inserting a new motion candidate into the HMVP table.

[0103] HMVP candidates may be used in the merge candidate list construction process. The latest several HMVP candidates in the table are checked in order and inserted into the candidate list after the TMVP candidate. Pruning may be applied on the HMVP candidates to the spatial or temporal merge candidate excluding sub-block motion candidate (i.e., ATMVP).

[0104] In some embodiments, to reduce the number of pruning operations, the number of HMPV candidates to be checked (denoted by L) is set as L = (N <=4) ? M: (8 - N), where N indicates a number of available non-sub-block merge candidates and M indicates a number of available HMVP candidates in the table. In addition, once the total number of available merge candidates reaches the signaled maximally allowed merge candidates minus 1, the merge candidate list construction process from the HMVP list is terminated. Furthermore, the number of pairs for combined bi-predictive merge candidate derivation is reduced from 12 to 6.

[0105] HMVP candidates could also be used in the AMVP candidate list construction process. The motion vectors of the last K HMVP candidates in the table are inserted after the TMVP candidate. Only HMVP candidates with the same reference picture as the AMVP target reference picture are used to construct the AMVP candidate list. Pruning is applied on the HMVP candidates. In some applications, K is set to 4 while the AMVP list size is kept unchanged, i.e., equal to 2.

[0106] According to some embodiments, when intra block copy operates as a separate mode from inter mode, a separate history buffer, referred to as HBVP, may be used for storing previously coded intra block copy block vectors. As a separate mode from inter prediction, it is desirable to have a simplified block vector derivation process for intra block copy mode. The candidate list for IBC BV prediction in AMVP mode may share the one used in IBC merge mode (merge candidate list), with 2 spatial candidates + 5 HBVP candidates.

[0107] The merge candidate list size of IBC mode may be assigned as MaxNumMergeCand. The MaxNumMergeCand may be determined by the inter mode merge candidate list size MaxNumMergeCand, which is specified, in some examples, as six_minus_max_num_merge_cand. The variable six_minus_max_num_merge_cand may specify the maximum number of merge motion vector prediction (MVP) candidates supported in a slice subtracted from 6.

[0108] In some examples, the maximum number of merging MVP candidates, MaxNumMergeCand is derived may be derived as:

$$\text{MaxNumMergeCand} = 6 - \text{six_minus_max_num_merge_cand}$$

[0109] The value of MaxNumMergeCand may be in the range of 1 to 6, inclusive. The BV prediction in non-merge mode may share the same list generated for IBC mege mode. However, in some examples, for non-merge mode case, the candidate list size is always 2. Accordingly, there is a need to develop proper methods to handle the IBC merge candidate list size as well as IBC non-merge mode (AMVP mode) predictor list size when MaxNumMergeCand is set to be various values and the maximum number of IBC merge candidate list is set differently compared to an inter merge candidate list size.

[0110] 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 some embodiments, the term block may be interpreted as a prediction block, a coding block, or a coding unit (i.e., CU). When merge mode is discussed, the skip mode may be regarded as a special merge mode. All disclosed embodiments for merge the mode may be applied to the skip mode as well.

[0111] In some embodiments, when MaxNumMergeCand for inter merge mode is signaled as 1, the corresponding merge candidate list size for IBC is 1, which may cause undesired behaviour, especially when IBC AMVP mode also uses the same candidate list for BV prediction. In this situation, a candidate list with at least two entries is expected (AMVP mode has two predictors). Emboidments of the present disclosure resolve these issues.

[0112] According to some embodiments, in a first method, the merge list size for IBC mode is at least M, where M is an integer number. In some embodiments, the merge list size for IBC, MaxNumIBCMergeCand, is set to be the following:

MaxNumIBCMergeCand = max(MaxNumMergeCand, M).

[0113] In one embodiment, M is set equal to 1 to guarantee at least 1 candidate for IBC merge mode. In another embodiment, M is set equal to 2 to guarantee at least 2 candidates for IBC merge mode. Furthermore, the IBC AMVP mode sharing the same candidate list as IBC merge mode can ensure that there will be two entries in the candidate list for BV predicton in AMVP mode. In another embodiment, when the IBC AMVP mode shares the same candidate list as the IBC merge mode, M is set to 1 if the current CU is coded in IBC merge mode, and M is set to be 2 when current CU is coded in IBC AMVP mode.

[0114] Accoring to some embodiments, in a second method, the index of IBC merge mode (i.e., merge_idx) is signaled only when the merge candidate list size for IBC is greater than 1. In this second method, the the merge candidate list size for IBC (i.e., MaxNumIBCMergeCand) is derived first, which infers that the variable MaxNumIBCMergeCand for IBC can be different from MaxNumMergeCand for inter mode.

[0115] In some embodiments, if only the merge candidate size for IBC is considered, then the following applies:

When general_merge_flag is true:

if (CuPredMode[x0][y0] == MODE_IBC) {	
if(MaxNumIBCMergeCand > 1)	
merge_idx[x0][y0]	ae(v)
}	

Table 1

[0116] In some embodiments, if MaxNumIBCMergeCand is set equal to MaxNumMergeCand−1, then the following applies:

When merge_flag is true:

if (CuPredMode[x0][y0] = = MODE_IBC) {	
if(MaxNumMergeCand > 2)	
merge_idx[x0][y0]	ae(v)
}	

Table 2

[0117] In some embodiments, if IBC AMVP mode shares the same candidate list as IBC merge mode, at least two entries in the candidate list for BV predicton in AMVP mode are expected. To support this condition, the following assignment can be made for the merge candidate list size for IBC:

MaxNumIBCMergeCand = max(MaxNumMergeCand, 2).

[0118] Since MaxNumIBCMergeCand >=2 is guaranteed in this embodiment, the signaling of merge_idx does not need to depend on the size of inter merge candidate list, MaxNumMergeCand. One example of syntax table is as follows:

if (CuPredMode[x0][y0] = = MODE_IBC) {	
merge_idx[x0][y0]	ae(v)
}	

Table 3

[0119] According to some embodiments, in a third method, a separate merge candidate list size for IBC is signaled, and the range of this size, MaxNumIBCMergeCand, is from at least M, where M is an integer number. In this third method, the the merge candidate list size for IBC, MaxNumIBCMergeCand, may be signaled from a separate syntax element. In one embodiment, the range of MaxNumIBCMergeCand is from 2 to MaxNumMergeCand.

[0120] The following is one embodiment of a signaling method of the third method. In some examples, MaxNumIBCMergeCand <= MaxNumMergeCand. At picture or slice level,

if(sps_ibc_enabled_flag && MaxNumMergeCand >= 2)	
max_num_merge_cand_minus_max_num_ibc_cand	ue(v)

Table 4

[0121] The variable **max_num_merge_cand_minus_max_num_ibc_cand** specifies, in some example, the maximum number of IBC merge mode candidates supported in the slice subtracted from MaxNumMergeCand. The maximum number of IBC merge mode candidates, MaxNumIBCMergeCand, may be derived as follows:

MaxNumIBCMergeCand =

MaxNumMergeCand − max_num_merge_cand_minus_max_num_ibc_cand.

[0122] When max_num_merge_cand_minus_max_num_ibc_cand is present, the value of MaxNumIBCMergeCand is in the range of 2 to MaxNumMergeCand, inclusive, in some examples. When max_num_merge_cand_minus_max_num_ibc_cand is not present, MaxNumIBCMergeCand is set to 0 in some examples. When MaxNumIBCMergeCand is equal to 0, IBC merge mode is not allowed for the current slice in some examples.

[0123] In some embodiments, at the CU level, if all the following conditions are true, MergeIBCFlag[x0][y0] is set equal to 1:

(i) sps_ibc_enabled_flag is equal to 1.

(ii) general_merge_flag[x0][y0] is equal to 1.

(iii)CuPredMode[x0][y0] is equal to MODE_IBC.

(iv)MaxNumIBCMergeCand is greater than or equal to 2.

if(MergeIBCFlag[x0][y0]) {	
merge_ibc_idx[x0][y0]	ae(v)
}	

Table 5

[0124] The variable merge_ibc_idx is the index for IBC merge candidate list, which ranges from 0 to MaxNumIBCMergeCand – 1 in some examples. If MaxNumIBCMergeCand is greater than MaxNumMergeCand, the above embodiment regarding the CU level does not hold true.

[0125] In some embodiments, skip_flag or general_merge_flag is signaled first. To make sure that they can be used properly. A constraint should be imposed such that these flags shall not be signaled or shall only be false when IBC mode is chosen and the merge candidate size is smaller than the desired minimum number M, such as M=2. In one embodiment, when MaxNumIBCMergeCand is smaller than 2, IBC merge mode or skip mode is not used. If a skip mode flag is signaled before a block level IBC flag, then a constraint may be imposed on the IBC flag signaling such that when skip_flag is true, IBC flag is not signaled and inferred to be false. In another example, if the skip mode flag is signaled before the block level IBC flag, the IBC flag is signaled, but is still inferred to be false.

[0126] In some embodiments, if the block level IBC flag is signaled before general_merge_flag, then a constraint is imposed on the general_merge_flag signaling such that when IBC flag is true, general_merge_flag is not be signaled and inferred to be false, or the

general_merge_flag is signaled but is inferred to be false. An example syntax table design is as follows:

if(cu_skip_flag[x0][y0] = = 0 && !(CuPredMode[x0][y0] = = MODE_IBC && MaxNumIBCMergeCand < 2))	
general_merge_flag[x0][y0]	ae(v)
if(general_merge_flag[x0][y0]) {	
merge_data(x0, y0, cbWidth, cbHeight)	
} else if () {	

Table 6

[0127] When CuPredMode[x0][y0] = = MODE_IBC and MaxNumIBCMergeCand < 2, general_merge_flag[x0][y0] is inferred to be 0 in some examples.

[0128] According to some embodiments, in a fourth method, the merge list size for IBC mode is at least M, and at most N, where M and N are integer numbers, and M < N. In this fourth method, the merge list size for IBC, MaxNumIBCMergeCand, may be set to the following:

MaxNumIBCMergeCand = max(M, min(MaxNumMergeCand, N)).

[0129] In one embodiment, M is set equal to 2, so there is at least 2 candidates for IBC merge mode. In another embodiment, N is set equal to 5, so there is at most 5 candidates for IBC merge mode.

[0130] According to some embodiments, in a fifth method, a max number of IBC predictor candidates MaxNumIBCCand is signaled in bitstreams, such as in a slice header. This number may be used by both IBC merge and IBC AMVP. A IBC merge index and a IBC AMVP predictor index may be signaled using the same syntax flag (candidate index) with a truncated unary code and (MaxNumIBCCand – 1) being a max value of the truncated unary code.

[0131] According to some embodiments, in a sixth embodiment, regardless of whether MaxNumIBCMergeCand is equal to MaxNumMergeCand, when MaxNumIBCMergeCand is equal to 1, the mvp idx (mvp_l0_flag) for the IBC AMVP mode is not signaled. Thefore, in accordance with the sixth method, if the BV predictor size is equal to 1 (determined by MaxNumIBCMergeCand=1), then there will be only 1 predictor candidate for the IBC AMVP mode. When this situation occurs, the index of this predictor does not need to be signaled.

[0132] The below syntax is an example of an embodiment according to method 6.

……..	
} else if (CuPredMode[x0][y0] = = MODE_IBC) {	
mvd_coding(x0, y0, 0, 0)	
if(MaxNumIBCMergeCand > 1)	
mvp_l0_flag[x0][y0]	ae(v)
if(sps_amvr_enabled_flag && (MvdL0[x0][y0][0] != 0 || MvdL0[x0][y0][1] != 0)) {	
amvr_precision_flag[x0][y0]	ae(v)
}	
} else {	
……	

Table 7

[0133] FIG. 12 illustrates an embodiment of a video decoding process performed by a video decoder such as video decoder (710). The process may start at step (S1200) where a coded video bitstream including a current picture is received. The process proceeds to step (S1202) to determine whether a current block in the current picture is coded in the IBC mode. If the current block is coded in the IBC mode, the process proceeds from step (S1202) to step (S1204) to determine a number of IBC predictor candidates for the current block. For example, the number of IBC predictor candidates can be determined in accordance with one of the first to sixth methods disclosed above. The process proceeds to step (S1206) where an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates is constructed. The process proceeds to step (S1208) where a block vector predictor is selected from the IBC predictor candidate list. The process proceeds to step (S1210) where a block vector associated with the current block is decoded using the block vector predictor. The process proceeds to step (S1212) where the current block is decoded in accordance with the block vector that was decoded in the previous step. For example, a block vector predictor from the IBC predictor candidate list may be selected, a block vector may be decoded using the selected block vector predictor, and IBC decoding is performed on the current block using the block vector. In this regard, a block vector associated with the selected candidate is used to point to another block in the current picture that is used for decoding the current block.

[0134] Returning to step (S1202), if the current block is not coded in the IBC mode, the process proceeds to step (S1214) where the current block is decoded in accordance with a coding mode of the current block. For example, the current block may be decoded based on the intra prediction mode or inter prediction mode. The process illustrated in FIG. 12 may end after steps (S1212) and (S1214) are completed.

[0135] 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. 13 shows a computer system (1300) suitable for implementing certain embodiments of the disclosed subject matter.

[0136] 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.

[0137] 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.

[0138] The components shown in FIG. 13 for computer system (1300) 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 (1300).

[0139] Computer system (1300) 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).

[0140] Input human interface devices may include one or more of (only one of each depicted): keyboard (1301), mouse (1302), trackpad (1303), touch screen (1310), data-glove (not shown), joystick (1305), microphone (1306), scanner (1307), camera (1308).

[0141] Computer system (1300) 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 (1310), data-glove (not shown), or joystick (1305), but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers (1309), headphones (not depicted)), visual output devices (such as screens (1310) 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).

[0142] Computer system (1300) can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (1320) with CD/DVD or the like media (1321), thumb-drive (1322), removable hard drive or solid state drive (1323), 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.

[0143] 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.

[0144] Computer system (1300) 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 (1349) (such as, for example USB ports of the computer system (1300)); others are commonly integrated into the core of the computer system (1300) 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 (1300) 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.

[0145] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (1340) of the computer system (1300).

[0146] The core (1340) can include one or more Central Processing Units (CPU) (1341), Graphics Processing Units (GPU) (1342), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (1343), hardware accelerators for certain tasks (1344), and so forth. These devices, along with Read-only memory (ROM) (1345), Random-access memory (1346), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (1347), may be connected through a system bus (1348). In some computer systems, the system bus (1348) 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 (1348), or through a peripheral bus (1349). Architectures for a peripheral bus include PCI, USB, and the like.

[0147] CPUs (1341), GPUs (1342), FPGAs (1343), and accelerators (1344) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (1345) or RAM (1346). Transitional data can also be stored in RAM (1346), whereas permanent data can be stored for example, in the internal mass storage (1347). 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 (1341), GPU (1342), mass storage (1347), ROM (1345), RAM (1346), and the like.

[0148] 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.

[0149] As an example and not by way of limitation, the computer system having architecture (1300), and specifically the core (1340) 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 (1340) that are of non-transitory nature, such as core-internal mass storage

(1347) or ROM (1345). The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core (1340). A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core (1340) 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 (1346) 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 (1344)), which can operate in place of or together with software to execute particular processes or particular parts of particular processes described herein. Reference to software can encompass logic, and vice versa, where appropriate. Reference to a computer-readable media can encompass a circuit (such as an integrated circuit (IC)) storing software for execution, a circuit embodying logic for execution, or both, where appropriate. The present disclosure encompasses any suitable combination of hardware and software.

Appendix A: Acronyms

JEM: joint exploration model

VVC: versatile video coding

BMS: benchmark set

MV: Motion Vector

HEVC: High Efficiency Video Coding

SEI: Supplementary Enhancement Information

VUI: Video Usability Information

GOPs: Groups of Pictures

TUs: Transform Units,

PUs: Prediction Units

CTUs: Coding Tree Units

CTBs: Coding Tree Blocks

PBs: Prediction Blocks

HRD: Hypothetical Reference Decoder

SNR: Signal Noise Ratio

CPUs: Central Processing Units

GPUs: Graphics Processing Units

CRT: Cathode Ray Tube

LCD: Liquid-Crystal Display

OLED: Organic Light-Emitting Diode

CD: Compact Disc

DVD: Digital Video Disc

ROM: Read-Only Memory

RAM: Random Access Memory

ASIC: Application-Specific Integrated Circuit

PLD: Programmable Logic Device

LAN: Local Area Network

GSM: Global System for Mobile communications

LTE: Long-Term Evolution

CANBus: Controller Area Network Bus

USB: Universal Serial Bus

PCI: Peripheral Component Interconnect

FPGA: Field Programmable Gate Areas

SSD: solid-state drive

IC: Integrated Circuit

CU: Coding Unit

[0150] 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.

[0151] (1) A method of video decoding includes receiving a coded video bitstream including a current picture; determining whether a current block included in the current picture is coded in intra block copy (IBC) mode; in response to the current block being coded in the IBC mode, determining a number of IBC predictor candidates associated with the current block; constructing an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates; selecting a block vector predictor from the IBC predictor candidate list;

decoding a block vector associated with the current block using the block vector predictor; and decoding the current block in accordance with the block vector.

[0152] (2) The method according to feature (1), in which the number of IBC predictor candidates is greater than or equal to M and less than or equal to N, and wherein M is 2.

[0153] (3) The method according to feature (2), in which N is 5.

[0154] (4) The method of feature (2), in which the number of IBC predictor candidates is equal to: max (M, min (MaxNumMergeCand, N)), in which MaxNumMergeCand is equal to a number of candidates in a merge mode list.

[0155] (5) The method according to any one of features (1) – (4), in which the number of IBC predictor candidates is signaled in the bitstream.

[0156] (6) The method according to feature (5), in which the number of IBC predictor candidates is signaled using a truncated unary code with the number of IBC predictor candidates minus one being the max value.

[0157] (7) The method according to any one of features (1) – (6), in which when the number of IBC predictor candidates is equal to 1, an index associated with selecting a IBC block vector predictor is not signaled.

[0158] (8) A video decoder for performing video decoding, including processing circuitry configured to: receive a coded video bitstream including a current picture, determine whether a current block included in the current picture is coded in intra block copy (IBC) mode, in response to the current block being coded in the IBC mode, determine a number of IBC predictor candidates associated with the current block, construct an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates, select a block vector predictor from the IBC predictor candidate list, decode a block vector associated with the current block using the block vector predictor, and decode the current block in accordance with the block vector.

[0159] (9) The video decoder according to feature (8), in which the number of IBC predictor candidates is greater than or equal to M and less than or equal to N, and wherein M is 2.

[0160] (10) The video decoder according to feature (9), in which N is 5.

[0161] (11) The video decoder according to feature (9), in which the number of IBC predictor candidates is equal to: max (M, min (MaxNumMergeCand, N)), in which MaxNumMergeCand is equal to a number of candidates in a merge mode list.

[0162] (12) The video decoder according to any one of features (8) – (11), in which the number of IBC predictor candidates is signaled in the bitstream.

[0163] (13) The video decoder according to feature (12), in which the number of IBC predictor candidates is signaled using a truncated unary code with the number of IBC predictor candidates minus one being the max value.

[0164] (14) The video decoder according to any one of features (8) – (13), wherein when the number of IBC predictor candidates is equal to 1, an index associated with selecting a IBC block vector predictor is not signaled.

[0165] (15) 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 including: receiving a coded video bitstream including a current picture; determining whether a current block included in the current picture is coded in intra block copy (IBC) mode; in response to the current block being coded in the IBC mode, determining a number of IBC predictor candidates associated with the current block; constructing an IBC predictor candidate list having a size corresponding to the number of IBC predictor candidates; selecting a block vector predictor from the IBC predictor candidate list; decoding a block vector associated with the current block using the block vector predictor; and decoding the current block in accordance with the block vector.

[0166] (16) The non-transitory computer readable medium according to feature (15), in which the number of IBC predictor candidates is greater than or equal to M and less than or equal to N, and wherein M is 2.

[0167] (17) The non-transitory computer readable medium according to feature (16), in which N is 5.

[0168] (18) The non-transitory computer readable medium according to feature (16), in which the number of IBC predictor candidates is equal to: max (M, min (MaxNumMergeCand, N)), in which MaxNumMergeCand is equal to a number of candidates in a merge mode list.

[0169] (19) The non-transitory computer readable medium of feature (15), in which the number of IBC predictor candidates is signaled in the bitstream.

[0170] (20) The non-transitory computer readable medium of feature (19), in which the number of IBC predictor candidates is signaled using a truncated unary code with the number of IBC predictor candidates minus one being the max value.

WHAT IS CLAIMED IS:

1. A method of video encoding, comprising:

determining whether a current block included in a current picture is to be coded in an intra block copy (IBC) mode;

in response to determining that the current block is to be coded in the IBC mode, determining a maximum number of IBC predictor candidates associated with the current block based on a maximum of (i) a predetermined integer M and (ii) a maximum number of candidates MaxNumMergeCand in a merge mode list, wherein the maximum number of IBC predictor candidates is equal to a value of max (M, MaxNumMergeCand);

encoding the determined maximum number of IBC predictor candidates associated with the current block in a coded video bitstream comprising the current block in the current picture for constructing an IBC predictor candidate list;

encoding at least a block vector predictor included in the IBC predictor candidate list in the coded video bitstream for predicting a block vector associated with the current block; and

generating the coded video bitstream including the current block based on the encoded block vector and the encoded maximum number of IBC predictor candidates.

2. The method according to claim 1, wherein the predetermined integer M is equal to 2.

3. The method according to claim 2, wherein the predetermined integer M is equal to 2 based on the current block being coded in an IBC Advanced Motion Vector Prediction (AMVP) mode.

4. The method according to claim 1, wherein signaling of a motion vector predictor index for the IBC mode in the generated coded video bitstream is not dependent on the maximum number of candidates MaxNumMergeCand.

5. The method according to claim 4, wherein the motion vector predictor index for the IBC mode is signaled in the generated coded video bitstream.

6. The method according to claim 1, wherein a motion vector predictor index for the IBC mode is not signaled in the generated coded video bitstream based on the maximum number of IBC predictor candidates being equal to 1.

7. The method according to claim 1, wherein the determining the maximum number of IBC predictor candidates comprises:

determining the maximum number of IBC predictor candidates by processing circuitry of a video encoder.

8. The method according to claim 1, further comprising:

indicating the maximum number of candidates MaxNumMergeCand by including a syntax element in the generated coded video bitstream based on a predetermined constant.

9. The method according to claim 8, wherein

the predetermined constant is 6;

the syntax element in the generated coded video bitstream is six_minus_max_num_merge_cand; and

the maximum number of candidates MaxNumMergeCand is indicated by (6 - six_minus_max_num_merge_cand).

10. An apparatus for performing video encoding, comprising:

processing circuitry configured to:

determine whether a current block included in a current picture is to be coded in an intra block copy (IBC) mode;

in response to determining that the current block is to be coded in the IBC mode, determine a maximum number of IBC predictor candidates associated with the current block based on a maximum of (i) a predetermined integer M and (ii) a maximum number of candidates MaxNumMergeCand in a merge mode list, wherein the maximum number of IBC predictor candidates is equal to a value of max(M, MaxNumMergeCand);

encode the determined maximum number of IBC predictor candidates associated with the current block in a coded video bitstream comprising the current block in the current picture for constructing an IBC predictor candidate list;

encode at least a block vector predictor included in the IBC predictor candidate list in the coded video bitstream for predicting a block vector associated with the current block; and

generate the coded video bitstream including the current block based on the encoded block vector and the encoded maximum number of IBC predictor candidates.

11. An apparatus for performing video encoding, comprising:

a memory, configured to store computer-executable instructions; and

a processor, when executing the computer-executable instructions stored in the memory, configured to cause the apparatus to implement the method of any one of claims 1 to 9.

12. A computer readable medium storing a bitstream that can be encoded by a method of video encoding, the method comprising:

determining whether a current block included in a current picture is to be coded in an intra block copy (IBC) mode;

in response to determining that the current block is to be coded in the IBC mode, determining a maximum number of IBC predictor candidates associated with the current block based on a maximum of (i) a predetermined integer M and (ii) a maximum number of candidates MaxNumMergeCand in a merge mode list, wherein the maximum number of IBC predictor candidates is equal to a value of max (M, MaxNumMergeCand);

encoding the determined maximum number of IBC predictor candidates associated with the current block in a coded video bitstream comprising the current block in the current picture for constructing an IBC predictor candidate list;

encoding at least a block vector predictor included in the IBC predictor candidate list in the coded video bitstream for predicting a block vector associated with the current block; and

generating the coded video bitstream including the current block based on the encoded block vector and the encoded maximum number of IBC predictor candidates.

13. A non-transitory computer-readable medium having instructions stored therein, which when executed by a processor in a video encoder causes the processor to execute the method of any one of claims 1 to 9.