METHOD AND APPARATUS FOR VIDEO CODING

INCORPORATION BY REFERENCE

[0001] This present application claims the benefit of priority to U.S. Patent Application No. 17/217,595, "METHOD AND APPARATUS FOR VIDEO CODING" filed on March 30, 2021, which claims the benefit of priority to U.S. Provisional Application No. 63/005,511, "METHOD OF SETTING NUMBER OF SUBBLOCK MERGING CANDIDATES" filed on April 6, 2020. 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.

SUMMARY

[0010] Aspects of the disclosure provide methods and apparatuses for video encoding/decoding. In some examples, an apparatus for video decoding includes receiving circuitry and processing circuitry. For example, processing circuitry determines a parameter based on prediction information decoded from a coded video bitstream. The parameter is in a range that depends on a flag indicative of an enable/disable status of subblock based temporal motion vector prediction. Then, the processing circuitry calculates a maximum number of candidates in the subblock based merge candidate lists based on the parameter, and reconstructs, in response to a current block in a subblock based prediction mode, samples of the current block based on a candidate selection from a constructed subblock based merge candidate list of the current block. The constructed subblock based merge candidate list of the current block is constrained by the maximum number of candidates in the subblock based merge candidate lists.

[0011] In some examples, the processing circuitry determines the maximum number of candidates in the subblock based merge candidate lists by subtracting the parameter from a default number. In an example, the default number is 5.

[0012] In some embodiments, an upper limit of the range depends on the flag indicative of the enable/disable status of subblock based temporal motion vector prediction.

[0013] In an example, the processing circuitry receives the parameter that is signaled in the coded video bitstream. In another example, the processing circuitry infers the parameter based on a default number and the flag indicative of the enable/disable status of subblock based temporal motion vector prediction in response to the parameter not being signaled in the coded video bitstream.

[0014] In some examples, the flag is indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level.

[0015] In some embodiments, the parameter is in a range that depends on a first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level and a second flag indicative of the enable/disable status of temporal motion vector prediction at a picture header (PH) level. In some examples, the processing circuitry can infer the parameter based on a default number, the first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at the SPS level and the second flag indicative of the enable/disable status of temporal motion vector prediction at the PH level in response to the parameter not being signaled in the coded video bitstream.

[0016] Aspects of the disclosure also provide a non-transitory computer-readable medium storing instructions which when executed by a computer for video decoding cause the computer to perform the method for video decoding.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] 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:

[0018] FIG. 1 is a schematic illustration of a current block and its surrounding spatial merge candidates in one example.

[0019] FIG. 2 is a schematic illustration of a simplified block diagram of a communication system (200) in accordance with an embodiment.

[0020] FIG. 3 is a schematic illustration of a simplified block diagram of a communication system (300) in accordance with an embodiment.

[0021] FIG. 4 is a schematic illustration of a simplified block diagram of a decoder in accordance with an embodiment.

[0022] FIG. 5 is a schematic illustration of a simplified block diagram of an encoder in accordance with an embodiment.

[0023] FIG. 6 shows a block diagram of an encoder in accordance with another embodiment.

[0024] FIG. 7 shows a block diagram of a decoder in accordance with another embodiment.

[0025] FIGs. 8A-8B show affine motion models.

[0026] FIG. 9 shows an example of affine motion vector field per subblock.

[0027] FIG. 10 shows an example for determining inherited control point motion vectors in the affine merge mode.

[0028] FIG. 11 shows an example of spatial neighbors and temporal neighbor according to some embodiments of the disclosure.

[0029] FIG. 12 shows a diagram illustrating motion vector usage for affine motion data inheritance in some examples.

[0030] FIG. 13 shows a schematic illustration of an example of a prediction refinement with optical flow method in accordance with an embodiment of the disclosure.

[0031] FIGs. 14-15 show an example of a subblock based temporal motion vector prediction (SbTMVP) process according to some embodiments of the disclosure.

[0032] FIG. 16 shows a syntax table example for sequence parameter set (SPS) in some examples.

[0033] FIG. 17 shows a syntax table example for picture header structure in some examples.

[0034] FIG. 18 shows a flow chart outlining a process example according to some embodiments of the disclosure.

[0035] FIG. 19 is a schematic illustration of a computer system in accordance with an embodiment.

DETAILED DESCRIPTION OF EMBODIMENTS

[0036] 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.

[0037] 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.

[0038] 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.

[0039] 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.

[0040] 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.

[0041] 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.

[0042] 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.

[0043] 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).

[0044] 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.

[0045] 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).

[0046] 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.

[0047] 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.

[0048] 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).

[0049] 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).

[0050] 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.

[0051] 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.

[0052] 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.

[0053] 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.

[0054] 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.

[0055] 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.

[0056] 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.

[0057] 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).

[0058] 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.

[0059] 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.

[0060] 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.

[0061] 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).

[0062] 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.

[0063] 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.

[0064] 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).

[0065] 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).

[0066] 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.

[0067] 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.

[0068] 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).

[0069] 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:

[0070] 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.

[0071] 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.

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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.

[0076] 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.

[0077] 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.

[0078] Further, a merge mode technique can be used in the inter-picture prediction to improve coding efficiency.

[0079] 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.

[0080] 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.

[0081] 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.

[0082] 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.

[0083] 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.

[0084] 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.

[0085] 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.

[0086] 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.

[0087] 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.

[0088] 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.

[0089] 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.

[0090] 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).

[0091] The inter decoder (780) is configured to receive the inter prediction information, and generate inter prediction results based on the inter prediction information.

[0092] The intra decoder (772) is configured to receive the intra prediction information, and generate prediction results based on the intra prediction information.

[0093] 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).

[0094] 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.

[0095] 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.

[0096] Aspects of the disclosure provide techniques in the field of inter prediction in advanced video codec. The techniques can be used to set the number of candidates in a candidate list that can be referred to as a subblock merge candidate list.

[0097] In various embodiments, for an inter-predicted CU, motion parameters including motion vectors, reference picture indices, reference picture list usage index, and/or other additional information can be used for inter-predicted sample generation. Inter prediction can include uni-prediction, bi-prediction, and/or the like. In uni-prediction, a reference picture list (e.g., a first reference picture list or a list 0 (L0) or a second reference picture list or a list 1 (L1)) can be used. In bi-prediction, both L0 and L1 can be used. The reference picture list usage index can indicate that the reference picture list(s) includes L0, L1, or L0 and L1.

[0098] The motion parameters can be signaled in an explicit or implicit manner. When a CU is coded with a skip mode, the CU can be associated with one PU and can include no significant residual coefficients (e.g., the residual coefficients are zero), no coded motion vector difference (MVD), or no reference picture index.

[0099] A merge mode can be used in which motion parameters for a current CU can be obtained from neighboring CUs, including spatial and temporal merge candidates, and optionally other merge candidates. The merge mode can be applied to an inter-predicted CU, and may be used for a skip mode. Alternatively, motion parameters can be explicit transmitted or signaled. For example, motion vectors, a corresponding reference picture index for each reference picture

list, a reference picture list usage flag, and other information can be signaled explicitly per each CU.

[0100] In some examples (e.g., VVC), one or more of the following inter prediction coding tools are used: (1) extended merge prediction, (2) merge mode with motion vector difference (MMVD), (3) symmetric MVD (SMVD) signaling, (4) affine motion compensated prediction, (5) subblock-based temporal motion vector prediction (SbTMVP), (6) adaptive motion vector resolution (AMVR), (7) motion field storage: 1/16th luma sample MV storage and 8x8 motion field compression, (8) bi-prediction with CU-level weight (BCW), (9) bi-directional optical flow (BDOF), (10) decoder side motion vector refinement (DMVR), (11) geometric partitioning mode (GPM), and (12) combined inter and intra prediction (CIIP).

[0101] According to an aspect of the disclosure, some inter prediction coding tools may operate based on a subblock based merge candidate list. In an example, the affine motion compensated prediction can be performed in an affine merge mode (that is also referred to as subblock based merge mode in some examples). In the affine merge mode, prediction can be performed based on an affine merge candidate list that is referred to as subblock based merge candidate list in some examples. In another example, the subblock-based temporal motion vector prediction (SbTMVP) can also operate based on the subblock based merge candidate list.

[0102] For affine motion compensated prediction, in some examples (e.g., HEVC), only translation motion model is applied for motion compensation prediction (MCP). The real world has many kinds of motion, e.g. zoom in/out, rotation, perspective motions and the other irregular motions. In some examples (e.g., VVC), a block-based affine transform motion compensation prediction is applied.

[0103] FIGs. 8A-8B show affine motion models. FIG. 8A shows the affine motion field of a block that is described by motion information of two control points CP0 and CP1 (4-parameter affine model) and FIG. 8B shows the affine motion field of a block that is described by three control points CP0, CP1 and CP2 (6-parameter affine model).

[0104] In some embodiments, the 4-parameter affine motion model, motion vector (mv_x, mv_y) at sample location (x, y) in a block can be derived as (Eq. 1), and the 6-parameter affine motion model, motion vector at sample location (x, y) in a block can be derived as (Eq. 2):

$$\begin{cases} mv_x = \frac{mv_{1x} - mv_{0x}}{W}x + \frac{mv_{1y} - mv_{0y}}{W}y + mv_{0x} \\ mv_y = \frac{mv_{1y} - mv_{0y}}{W}x + \frac{mv_{1y} - mv_{0x}}{W}y + mv_{0y} \end{cases} \quad \text{(Eq. 1)}$$

$$\begin{cases} mv_x = \frac{mv_{1x} - mv_{0x}}{W}x + \frac{mv_{2x} - mv_{0x}}{H}y + mv_{0x} \\ mv_y = \frac{mv_{1y} - mv_{0y}}{W}x + \frac{mv_{2y} - mv_{0y}}{H}y + mv_{0y} \end{cases} \quad \text{(Eq. 2)}$$

where mv_0 or (mv_{0x}, mv_{0y}) denotes the motion vector of the top-left corner control point CP0, mv_1 or (mv_{1x}, mv_{1y}) is motion vector of the top-right corner control point CP1, mv_2 or (mv_{2x}, mv_{2y}) is motion vector of the bottom-left corner control point CP2, W denotes the width of the block and H denotes the height of the block.

[0105] In order to simplify the motion compensation prediction, block based affine transform prediction is applied.

[0106] FIG. 9 shows an example of affine MV field per subblock. In an example, a current CU 910 (e.g., of 16x16 luma samples) is divided into 4×4 luma subblocks (each subblock can be 4x4 luma samples). To derive motion vector of each 4×4 luma subblock, the motion vector of the center sample of each subblock, as shown in FIG 9, is calculated according to above equations (Eq.1) and (Eq. 2). The motion vector can be rounded to 1/16 fraction accuracy for example. Then the motion compensation interpolation filters are applied to generate the prediction of each subblock with derived motion vector. In some examples, the subblock size of chroma-components can be set to 4×4 as well, and thus a 4×4 chroma subblock includes four corresponding 4×4 luma subblocks. The MV of a 4×4 chroma subblock is calculated as the average of the MVs of the four corresponding 4×4 luma subblocks in an example.

[0107] It is noted that subblocks can be defined to have other suitable number of luma samples. It is also noted that subblocks are referred to as sub-CUs in some examples.

[0108] For translational motion inter prediction, two affine motion inter prediction modes, that are referred to as affine merge (AF_MERGE) mode and affine advanced MVP (affine AMVP) mode, can be used.

[0109] For affine merge prediction, in an example, AF_MERGE mode can be applied for CUs with both width and height larger than or equal to 8. In the AF_MERGE mode, the control point motion vectors (CPMVs) of the current CU are generated based on the motion information of the spatial neighboring CUs. In an example, an affine merge candidate list (also referred to as subblock based merge candidate list) can include up to five control point motion vector predictor (CPMVP) candidates and an index is signalled to indicate the one to be used for the current CU. In an example, three types of CPMVP candidates are used to form the affine merge candidate list. The first type of CPMVP candidates is inherited affine merge candidates that extrapolated

from the CPMVs of the neighbour CUs. The second type of CPMVP candidates are constructed affine merge candidates CPMVPs that are derived using the translational MVs of the neighbour CUs. The third type of CPMVP candidates uses Zero MVs.

[0110] In some examples, such as in VVC, a maximum of two inherited affine candidates can be used. In an example, two inherited affine candidates are derived from affine motion models of the neighboring blocks, one from left neighboring CUs (referred to as left predictor) and one from above neighboring CUs (referred to as above predictor). Using the neighboring blocks shown in FIG. 1 as an example, for the left predictor, the scan order is A0->A1, and for the above predictor, the scan order is B0->B1->B2. In an example, only the first inherited candidate available from each side is selected. In some examples, no pruning check is performed between two inherited candidates. When a neighboring affine CU is identified, the control point motion vectors of the neighboring affine CU are used to derive a CPMVP candidate in the affine merge candidate list of the current CU.

[0111] FIG. 10 shows an example for determining inherited control point motion vectors in the affine merge mode. As shown in FIG. 10, when the neighbour left bottom subblock A is coded in affine mode, the motion vectors mv_2 , mv_3 and mv_4 of the top left corner, above right corner and left bottom corner of a CU which contains the subblock A can be obtained. When subblock A is coded with 4-parameter affine model, the two CPMVs of the current CU are calculated according to mv_2, and mv_3. In case that subblock A is coded with 6-parameter affine model, the three CPMVs of the current CU are calculated according to mv_2 , mv_3 and mv_4.

[0112] In some examples, a constructed affine candidate is constructed by combining the neighbor translational motion information of each control point. The motion information for the control points can be derived from the specified spatial neighbors and temporal neighbor.

[0113] FIG. 11 shows an example of spatial neighbors (e.g., subblocks A0-A2 and B0-B3) and temporal neighbor (e.g., T) according to some embodiments of the disclosure. In an example, $CPMV_k$ (k=1, 2, 3, 4) represents the k-th control point. For $CPMV_1$, the B2->B3->A2 blocks are checked (-> is used for checking order) and the MV of the first available block is used as $CPMV_1$. For $CPMV_2$, the B1->B0 blocks are checked and the MV of the first available block is used as $CPMV_2$. For $CPMV_3$, the A1->A0 blocks are checked, and the MV of the first available block is used as $CPMV_3$. For TMVP, T is checked and is used as $CPMV_4$ if the MV of the block T is available.

[0114] After MVs of four control points $CPMV_1$-$CPMV_4$ are obtained, affine merge candidates are constructed based on that motion information. The following combinations of control point MVs are used to construct in order: {$CPMV_1$, $CPMV_2$, $CPMV_3$}, {$CPMV_1$, $CPMV_2$, $CPMV_4$}, {$CPMV_1$, $CPMV_3$, $CPMV_4$},{$CPMV_2$, $CPMV_3$, $CPMV_4$}, {$CPMV_1$, $CPMV_2$}, {$CPMV_1$, $CPMV_3$}.

[0115] The combination of 3 CPMVs can construct a 6-parameter affine merge candidate and the combination of 2 CPMVs can construct a 4-parameter affine merge candidate. In an example, to avoid motion scaling process, when the reference indices of control points are different, the related combination of control point MVs can be discarded.

[0116] In an example, after inherited affine merge candidates and constructed affine merge candidate are checked, if a candidate list is still not full, zero MVs are inserted to the end of the list.

[0117] For affine AMVP prediction, the affine AMVP mode can be applied on CUs with both width and height larger than or equal to 16. In some examples, an affine flag at CU level is signalled in the bitstream (e.g., coded video bitstream) to indicate whether affine AMVP mode is used in the CU and then another flag is signaled to indicate whether 4-parameter affine or 6-parameter affine is used. In the affine AMVP mode, the difference of the CPMVs of current CU and their predictors CPMVPs can be signalled in the bitstream. The affine AMVP candidate list size is 2 and the affine AMVP candidate list is generated by using the following four types of CPMV candidate in the order: (1) inherited affine AMVP candidates that extrapolated from the CPMVs of the neighbour CUs; (2) constructed affine AMVP candidates CPMVPs that are derived using the translational MVs of the neighbour CUs; (3) translational MVs from neighboring CUs; and (4) Zero MVs.

[0118] In some examples, the checking order of inherited affine AMVP candidates is the same as the checking order of inherited affine merge candidates. In an example, the only difference between the affine merge prediction and affine AMVP prediction is that, for AMVP candidate, only the affine CU that has the same reference picture as the current block is considered. In an example, no pruning process is applied when inserting an inherited affine motion predictor into the candidate list.

[0119] In some examples, constructed AMVP candidate can be derived from the specified spatial neighbors shown in FIG. 11. In an example, the same checking order is used as done in the candidate construction for the affine merge prediction. In addition, reference picture

index of the neighboring block is also checked. The first block in the checking order that is inter coded and has the same reference picture as in current CUs is used. When the current CU is coded with 4-parameter affine mode, and motion vectors of two control points (e.g., {$CPMV_1$, $CPMV_2$}) are both available, the motion vectors of the two control points are added as one candidate in the affine AMVP list. When the current CU is coded with 6-parameter affine mode, and all three motion vectors of the control points CPMVs (e.g., {$CPMV_1$, $CPMV_2$, $CPMV_3$}) are available, they are added as one candidate in the affine AMVP list. Otherwise, constructed AMVP candidate is set as unavailable.

[0120] When the number of affine AMVP list candidates is still less than 2 after inherited affine AMVP candidates and constructed AMVP candidate are checked, $CPMV_1$, $CPMV_2$ and $CPMV_3$, will be added, in order, as the translational MVs to predict all control point MVs of the current CU, when available. Finally, zero MVs are used to fill the affine AMVP list if the affine AMVP list is still not full.

[0121] According to some aspects of the disclosure, motion information can be stored in suitable buffers, such as a local buffer, a picture line buffer and the like. The local buffer is used to store motion information at CTU level, such as motion vectors of 4×4 blocks in a CTU. For example, when a CU in a CTU is decoded based on inter prediction, motion vectors of respective 4×4 blocks of the CU can be stored in the local buffer and can be used to decode later CUs. The picture line buffer is used to store motion information of CTUs that are above the current CTU, such as motion vectors of 4×4 blocks at the bottom of the above CTUs. The above CTUs of the current CTU can be referred to as above CTU line.

[0122] In some examples (e.g., VVC), the CPMVs of affine CUs are stored separately from the motion vectors of 4×4 blocks. In an example, the local buffer includes a first portion to store the motion vectors of 4×4 blocks in a CTU, and a second portion to store the CPMVs of affine CUs in the CTU. The stored CPMVs in the second portion of the local buffer can be used to generate the inherited CPMVPs in affine merge mode and affine AMVP mode for the lately coded CUs. The subblock MVs derived from CPMVs are used for motion compensation, MV derivation of merge/AMVP list of translational MVs and de-blocking.

[0123] In some embodiments, the picture line buffer does not store the additional CPMVs of affine CUs in the above CTU line. In some examples, affine motion data inheritance from the CUs from above CTUs is treated differently to the inheritance from the normal neighboring CUs in the same CTU line. When the candidate CU for affine motion data

inheritance is in the above CTU line, the bottom-left and bottom-right subblock MVs in the picture line buffer, instead of the CPMVs, are used for the affine MVP derivation. Thus, the CPMVs are only stored in local buffer not in the picture line buffer in some examples. In an example that the candidate CU is 6-parameter affine coded, the affine model can be degraded to 4-parameter model.

[0124] FIG. 12 shows a diagram illustrating motion vector usage for affine motion data inheritance in some examples. In FIG. 12, each small square represents a subblock of 4×4, and motion vector of a subblock can be the motion vector at the center of the subblock. Further, the current CU is located at a top position of a current CTU. As shown in FIG. 12, in the neighboring CUs of the current CU, CU-E and CU-D are affine coded. CU-D is in the same CTU line as the current CU, and CU-E is in the above CTU line of the current CU. The CPMVs of CU-D can be stored in the local buffer. For example, for 4-parameter affine model, mv_{D0} and mv_{D1} are stored in the local buffer, and CPMVs (e.g., mv_0 and mv_1) of the current CU can be calculated according to mv_{D0} and mv_{D1} and the corresponding locations of the control points for mv_{D0} and mv_{D1}.

[0125] In an example, the picture line buffer stores the motion vectors of subblocks that are at the bottom of the above CTU line. CPMVs of CU-E, such as shown by mv_{E0} and mv_{E1} are not stored in the picture line buffer. In an example, the motion vectors of bottom-left subblock and bottom-right subblock of CU-E, such as shown by mv_{LE0} and mv_{LE1} are used for affine inheritance of the current CU. For example, CPMVs (e.g., mv_0 and mv_1) of the current CU can be calculated according to mv_{LE0} and mv_{LE1} and corresponding center locations of the two subblocks.

[0126] In some embodiments, a prediction refinement with optical flow (PROF) (also referred to as a PROF method) may be implemented to improve the subblock based affine motion compensation to achieve a finer granularity of motion compensation without increasing the memory access bandwidth for motion compensation. In an embodiment (e.g., VVC), after the subblock based affine motion compensation is performed, differences (or refinement values, refinements, prediction refinements) derived based on an optical flow equation can be added to predicted samples (e.g., luma predicted samples, or luma prediction samples) to obtain refined predicted samples.

[0127] FIG. 13 shows a schematic illustration of an example of the PROF method in accordance with an embodiment of the disclosure. A current block (1310) can be divided into

four subblocks (1312, 1314, 1316, and 1318). Each of the subblocks (1312, 1314, 1316, and 1318) can have a size of 4×4 pixels or samples. A subblock MV (1320) for the subblock (1312) can be derived according to CPMVs of the current block 1310, for example, using a center location of the subblock (1312) and an affine motion model (e.g., the 4-parameter affine motion model, the 6-parameter affine motion model). The subblock MV (1320) can point to a reference subblock (1332) in a reference picture. Initial subblock predicted samples can be determined according to the reference subblock (1332).

[0128] In some examples, a translational motion from the reference subblock (1332) to the subblock (1312) as described by the subblock MV (1320) may not predict the subblock (1312) with a high accuracy. In addition to the translational motion described by the subblock MV (1320), the subblock (1312) can also experience a non-translational motion (e.g., a rotation as seen in FIG. 13). Referring to FIG. 13, a subblock (1350) in the reference picture having shaded samples (e.g., a sample (1332a)) corresponds to and can be used to reconstruct the samples in the subblock (1312). The shaded sample (1332a) can be shifted by the pixel MV (1340) to reconstruct a sample (1312a) in the subblock (1312) with a high accuracy. Thus, in some examples, when non-translational motion occurs, to improve an accuracy of the prediction, a suitable prediction refinement method can be applied in an affine motion model, as described below.

[0129] In an example, the PROF method is implemented using the following four steps. In Step (1), the subblock based affine motion compensation can be performed to generate a prediction, such as an initial subblock prediction $I(i, j)$, for a current subblock (e.g., the subblock (1312)) where i and j are coordinates corresponding to a sample at position (i, j) (also referred to as a sample position, a sample location) in the current subblock (1312).

[0130] In Step (2), gradient calculations can be performed where spatial gradients $g_x(i,j)$ and $g_y(i,j)$ of the initial subblock prediction $I(i, j)$ at each sample position (i, j) can be calculated using, for example, a 3-tap filter [-1, 0, 1] according to Eq. 3 and Eq. 4 as below:

$$g_x(i, j) = I(i + 1, j) - I(i - 1, j) \quad \text{(Eq. 3)}$$

$$g_y(i, j) = I(i, j + 1) - I(i, j - 1) \quad \text{(Eq. 4)}$$

The subblock prediction can be extended by one pixel on each side for the gradient calculations. In some embodiments, to reduce the memory bandwidth and complexity, the pixels on the extended borders can be copied from the nearest integer pixel position in the reference picture

(e.g., the reference picture including the subblock (1332)). Therefore, additional interpolation for a padding region can be avoided.

[0131] In Step (3), a prediction refinement $\Delta I(i,j)$ can be calculated by Eq. 5 (e.g., an optical flow equation) as below.

$$\Delta I(i,j) = g_x(i,j) \times \Delta mv_x(i,j) + g_y(i,j) \times \Delta mv_y(i,j) \quad \text{(Eq. 5)}$$

where $\Delta mv(i,j)$ (e.g., ΔMV (1342)) is a difference MV between a pixel MV or a sample MV $mv(i,j)$ (e.g. pixel MV (1340)) for a sample location (i,j) and the subblock MV Mv_{SB} (e.g., the subblock MV (1320)) of the subblock (e.g., the subblock (1312)) where the sample location (i,j) is located. $\Delta mv(i,j)$ can also be referred to as a MV refinement (MVR) for the sample that is at the sample location (i, j) or the sample (i, j). $\Delta mv(i,j)$ can be determined using Eq. 6 as below.

$$\Delta mv(i,j) = mv(i,j) - mv_{SB} \quad \text{(Eq. 6)}$$

$\Delta mv_x(i,j)$ and $\Delta mv_y(i,j)$ are an x component (e.g., a horizontal component) and a y component (e.g., a vertical component) of the difference MV $\Delta mv(i,j)$, respectively.

[0132] Since the affine model parameters and the pixel locations relative to the subblock center position are not changed from one subblock to another subblock, $\Delta mv(i,j)$ can be calculated for a first subblock (e.g., the subblock (1312)), and reused for other subblocks (e.g., the subblocks (1314), (1316), and (1318)) in the same current block (1310). In some examples, x and y represent a horizontal shift and a vertical shift of a sample position (i,j) with respect to the center position of the subblock (1312), $\Delta mv(i,j)$ (e.g., including $\Delta mv_x(i,j)$ and $\Delta mv_y(i,j)$) can be derived by Eq. 7 as below,

$$\begin{cases} \Delta mv_x(x,y) = a \times x + b \times y \\ \Delta mv_y(x,y) = c \times x + d \times y \end{cases} \quad \text{(Eq. 7)}$$

where $\Delta mv_x(x,y)$ is the x component $\Delta mv_x(i,j)$ and $\Delta mv_y(x,y)$ is the y component $\Delta mv_y(i,j)$.

[0133] In an example, for a 4-parameter affine motion model, the parameters a-d are described by (Eq. 1). For a 6-parameter affine motion model, the parameters a-d are described by (Eq. 2) as described above.

[0134] In Step (4), the prediction refinement $\Delta I(i,j)$ (e.g., the luma prediction refinement) can be added to the initial subblock prediction I(i, j) to generate another prediction, such as a refined prediction $I'(i,j)$. The refined prediction $I'(i,j)$ can be generated using Eq. 8 as below for the sample (i, j):

$$I'(i,j) = I(i,j) + \Delta I(i,j). \quad \text{(Eq. 8)}$$

[0135] In some cases, PROF is not applied for an affine coded CU. In an example, all control point MVs are the same, which indicates the CU only has translational motion, and PROF is not applied. In another example, the affine motion parameters are greater than a specified limit then PROF is applied. In the second case, the subblock based affine motion compensation is degraded to CU based motion compensation to avoid large memory access bandwidth requirement.

[0136] In some embodiments, a fast encoding method can be applied to reduce the encoding complexity of affine motion estimation with PROF. In the fast encoding method, PROF is not applied at affine motion estimation stage in following two situations. In a first situation, if a current CU is not the root block and its parent block does not select the affine mode as its best mode, PROF is not applied since the possibility for the current CU to select the affine mode as best mode is low. In a second situation, if the magnitude of four affine parameters (a-d) are all smaller than a predefined threshold and the current picture is not a low delay picture, PROF is not applied because the improvement introduced by PROF is small for this situation. In this way, the affine motion estimation with PROF can be accelerated.

[0137] In some examples (e.g., VVC) , the subblock based temporal motion vector prediction (SbTMVP) can be used. Similar to the temporal motion vector prediction (TMVP) in HEVC, SbTMVP uses the motion field in the collocated picture to improve motion vector prediction and merge mode for CUs in the current picture. In some examples, the same collocated picture used by TMVP is used for SbTMVP. SbTMVP differs from TMVP in two aspects. In the first aspect, TMVP predicts motion at CU level but SbTMVP predicts motion at sub-CU level. In the second aspect, TMVP fetches the temporal motion vectors from the collocated block in the collocated picture (the collocated block is the bottom-right or center block relative to the current CU), SbTMVP applies a motion shift before fetching the temporal motion information from the collocated picture. The motion shift is obtained from the motion vector from one of the spatial neighboring blocks of the current CU.

[0138] FIGs. 14-15 show an example of a SbTMVP process according to some embodiments of the disclosure. SbTMVP predicts the motion vectors of the sub-CUs within the current CU in two steps. In the first step, the spatial neighbor A1 shown in FIG. 14 is examined. If the spatial neighbor A1 has a motion vector that uses the collocated picture as its reference picture, the motion vector is selected to be the motion shift to be applied. If no such motion is identified, then the motion shift is set to (0, 0).

[0139] In the second step, the motion shift identified in the first step is applied (i.e. added to the current block's coordinates) to obtain sub-CU-level motion information (motion vectors and reference indices) from the collocated picture as shown in FIG. 15. In the FIG. 15 example, A1's motion vector is set as the motion shift (1510). Then, for each sub-CU, the motion information of the corresponding block (the smallest motion grid that covers the center sample) in the collocated picture is used to derive the motion information for the sub-CU. After the motion information of the collocated sub-CU is identified, it is converted to the motion vectors and reference indices of the current sub-CU in a similar way as the TMVP process of HEVC. For example, temporal motion scaling is applied to align the reference pictures of the temporal motion vectors to those of the current CU.

[0140] In some examples, such as in VVC, a subblock based merge candidate list is used for the signaling of subblock based merge mode. The subblock based merge candidate list can include both SbTMVP candidate and affine merge candidates and is referred to as a combined subblock based merge candidate list in some examples. The SbTMVP mode is enabled/disabled by a flag, such as a sequence parameter set (SPS) flag. When the SbTMVP mode is enabled, in an example, the SbTMVP predictor is added as the first entry of the combined subblock based merge candidate list, and followed by the affine merge candidates. In some examples (e.g., VVC), the maximum allowed size of the combined subblock based merge candidate list is 5. It is noted that the maximum allowed size of the combined subblock based merge candidate list can be other suitable number.

[0141] In an example, the sub-CU size used in SbTMVP is fixed to be 8x8, and as done for affine merge mode, SbTMVP mode is only applicable to the CU with both width and height are larger than or equal to 8.

[0142] In some embodiments, the encoding logic of the additional SbTMVP merge candidate is the same as for the other merge candidates. In an example, for each CU in P or B slice, an additional rate distortion check is performed to decide whether to use the SbTMVP candidate.

[0143] According to some aspects of the disclosure, the maximum number of candidates in the combined subblock based merge candidate list can be signaled.

[0144] FIG. 16 shows a syntax table example (1600) for sequence parameter set (SPS) in some examples. The SPS includes information that may be applied to a series of consecutive coded video pictures (also referred to as a coded video sequence)

[0145] In the syntax table example (1600), a flag sps_temporal_mvp_enabled_flag is signaled as shown by (1610). The flag sps_temporal_mvp_enabled_flag equal to 1 specifies that temporal motion vector predictors may be used in the coded video; and the flag sps_temporal_mvp_enabled_flag equal to 0 specifies that temporal motion vector predictors are not used in the coded video. In some examples, the coded video can be referred to as a coded layer video sequence (CLVS), which is a group of pictures belonging to the same layer, starting with a random access point, and followed by pictures that may depend on each other and the random access point picture.

[0146] In related syntax table example (1600), when the flag sps_temporal_mvp_enabled_flag is equal to 1, two flags sps_sbtmvp_enabled_flag and sps_affine_enabled_flag may be signaled, as shown by (1620) and (1630). The flag sps_sbtmvp_enabled_flag equal to 1 specifies that subblock based temporal motion vector predictors may be used in decoding of pictures with slices having slice type not equal to I (intra coded) in the coded video. The flag sps_sbtmvp_enabled_flag equal to 0 specifies that subblock based temporal motion vector predictors are not used in the coded video. In an example, when the flag sps_sbtmvp_enabled_flag is not signaled, the flag sps_sbtmvp_enabled_flag can be inferred to be equal to 0.

[0147] The flag sps_affine_enabled_flag specifies whether affine model based motion compensation can be used for inter prediction. When the flag sps_affine_enabled_flag is equal to 0, in some examples, the syntax is constrained such that no affine model based motion compensation is used in the coded video. Otherwise (sps_affine_enabled_flag is equal to 1), affine model based motion compensation can be used in the coded video.

[0148] In the syntax table example (1600), when the flag sps_affine_enabled_flag is equal to 1, a parameter, such as five_minus_max_num_subblock_merge_cand, may be signaled. The parameter five_minus_max_num_subblock_merge_cand specifies the maximum number of subblock based merge candidates supported in the SPS subtracted from 5. The value of five_minus_max_num_subblock_merge_cand is in the range of 0 to 5, inclusive in some examples. For example, when the value of five_minus_max_num_subblock_merge_cand is 2, the maximum number of candidates in a combined subblock based merge candidate list is 3 (5 subtracts 2).

[0149] In some examples, the temporal motion vector predictor can be enabled/disabled at picture header level. FIG. 17 shows a syntax table example (1700) for picture header structure in some examples.

[0150] In the syntax table example (1700), when the SPS level flag sps_temporal_mvp_enabled_flag is equal to 1, a flag ph_temporal_mvp_enabled_flag is signaled as shown by (1710). The flag ph_temporal_mvp_enabled_flag specifies whether temporal motion vector predictors can be used for inter prediction for slices associated with the picture header. If ph_temporal_mvp_enabled_flag is equal to 0, the syntax elements of the slices associated with the picture header are constrained such that no temporal motion vector predictor is used in decoding of the slices. Otherwise (ph_temporal_mvp_enabled_flag is equal to 1), temporal motion vector predictors may be used in decoding of the slices associated with the picture header. When not present, in an example, the value of ph_temporal_mvp_enabled_flag is inferred to be equal to 0. When no reference picture in a decoded picture buffer has the same spatial resolution as the current picture, the value of ph_temporal_mvp_enabled_flag shall be equal to 0.

[0151] The maximum number of subblock based merge candidates can be derived based on flags and parameters that are signaled or inferred. In an example, a variable MaxNumSubblockMergeCand is used to denote the maximum number of subblock based merge candidates. In an example, when sps_affine_enabled_flag is equal to 1, MaxNumSubblockMergeCand is derived according to (Eq. 9), and when sps_affine_enabled_flag is equal to 0, MaxNumSubblockMergeCand is derived according to (Eq. 10):

$$MaxNumSubblockMergeCand = 5 - five_minus_max_num_subblock_merge_cand \quad (Eq.\ 9)$$

$$MaxNumSubblockMergeCand = sps_sbtmvp_enabled_flag \;\&\&\; ph_temporal_mvp_enabled_flag \quad (Eq.\ 10)$$

[0152] In some examples, the value of MaxNumSubblockMergeCand is in the range of 0 to 5, inclusive.

[0153] According to an aspect of the disclosure, when sps_affine_enabled_flag is signaled as 1, the MaxNumSubblockMergeCand is derived from five_minus_max_num_subblock_merge_cand as described in (Eq. 9). In some examples, a scenario that sps_affine_enabled_flag is signaled as 1 and

five_minus_max_num_subblock_merge_cand is signaled to be equal to 5 is allowed. In this scenario, the maximum number of subblock based merging candidates MaxNumSubblockMergeCand is derived as 0 which can turn off the affine merge mode as well as for SbTMVP regardless of the SbTMVP enabling flags, and may cause conflicts when SbTMVP enabling flags are indicative of sbTMVP being enabled.

[0154] Aspects of the disclosure provide techniques to set the value range of the number for subblock based merge candidates (also referred to as maximum number of subblock based merge candidates) according to a default number (e.g. denoted by N) of subblock based merge candidates and related high-level usage flag for affine and/or SbTMVP coding tools. For example, when the SbTMVP enabling flags are indicative of SbTMVP being enabled, the maximum number of subblock based merge candidates is not 0.

[0155] In some embodiments, the parameter five_minus_max_num_subblock_merge_cand has a negative correlation with the maximum number of subblock based merge candidates, and the upper limit of the parameter five_minus_max_num_subblock_merge_cand is determined based on the SbTMVP enabling flags.

[0156] In an embodiment, the parameter five_minus_max_num_subblock_merge_cand specifies the maximum number of subblock-based merging motion vector prediction candidates supported in the SPS subtracted from N, and further the value of the parameter five_minus_max_num_subblock_merge_cand is constrained in the range of 0 to N - sps_sbtmvp_enabled_flag, inclusive. The upper limit of the parameter five_minus_max_num_subblock_merge_cand depends on the flag sps_sbtmvp_enabled_flag.

[0157] In some examples, the default number N is 5, when the flag sps_sbtmvp_enabled_flag is 0 (SbTMVP is disabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be in the range of 0 to 5 inclusive. However, when the flag sps_sbtmvp_enabled_flag is 1 (SbTMVP is enabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be in the range of 0 to 4 inclusive. In an example, at the encoder side, when the flag sps_sbtmvp_enabled_flag is 1 and a calculated value of the parameter five_minus_max_num_subblock_merge_cand is 5 that is above the upper limit, a signaled value of the parameter five_minus_max_num_subblock_merge_cand in the coded video bitstream is 4 that is constrained in the range of 0 to 4 inclusive.

[0158] In some examples, when the value of the parameter five_minus_max_num_subblock_merge_cand is equal to the upper limit of the range, the parameter five_minus_max_num_subblock_merge_cand may not be signaled in the coded video bitstream from the encoder side. At the decoder side, when the decoder detects that the parameter five_minus_max_num_subblock_merge_cand is not present in the coded video bitstream, the decoder can infer the value of the parameter five_minus_max_num_subblock_merge_cand to be the upper limit of the range. The upper limit of the range can be determined based on the SbTMVP enabling flags. For example, the value of five_minus_max_num_subblock_merge_cand is inferred to be equal to N - sps_sbtmvp_enabled_flag. In an example, the default number N is 5, when the flag sps_sbtmvp_enabled_flag is 0 (SbTMVP is disabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be inferred to be 5. However, when the flag sps_sbtmvp_enabled_flag is 1 (SbTMVP is enabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be inferred to be 4.

[0159] In another embodiment, the upper limit of the parameter five_minus_max_num_subblock_merge_cand is determined based on a combination of multiple SbTMVP enabling flags, such as a first flag sps_sbtmvp_enabled_flag at the SPS level and a second flag ph_temporal_mvp_enabled_flag at the picture header level. In an example, the value of the parameter five_minus_max_num_subblock_merge_cand is constrained in the range of 0 to N - (sps_sbtmvp_enabled_flag && ph_temporal_mvp_enabled_flag), inclusive. When the parameter five_minus_max_num_subblock_merge_cand is not present in the coded video bitstream, the value of five_minus_max_num_subblock_merge_cand is inferred to be equal to N - (sps_sbtmvp_enabled_flag && ph_temporal_mvp_enabled_flag).

[0160] In some examples, the default number N is 5, when at least one of the first flag sps_sbtmvp_enabled_flag and the second flag ph_temporal_mvp_enabled_flag is 0 (SbTMVP is disabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be in the range of 0 to 5 inclusive. However, when both of the first flag sps_sbtmvp_enabled_flag and the second flag ph_temporal_mvp_enabled_flag are 1 (SbTMVP is enabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be in the range of 0 to 4 inclusive. In an example, at the encoder side, when both of the first flag sps_sbtmvp_enabled_flag and the second flag ph_temporal_mvp_enabled_flag are 1, and a calculated value of the parameter five_minus_max_num_subblock_merge_cand is 5 that is above the upper limit, a signaled value

of the parameter five_minus_max_num_subblock_merge_cand in the coded video bitstream is 4 that is constrained in the range of 0 to 4 inclusive.

[0161] In some examples, when the value of the parameter five_minus_max_num_subblock_merge_cand is equal to the upper limit of the range, the parameter five_minus_max_num_subblock_merge_cand may not be signaled in the coded video bitstream from the encoder side. At the decoder side, when the decoder detects that the parameter five_minus_max_num_subblock_merge_cand is not present in the coded video bitstream, the decoder can infer the value of the parameter five_minus_max_num_subblock_merge_cand to be the upper limit of the range. The upper limit of the range can be determined based on, for example, a suitable combination of the first flag sps_sbtmvp_enabled_flag and the second flag ph_temporal_mvp_enabled_flag. For example, the value of five_minus_max_num_subblock_merge_cand is inferred to be equal to N - (sps_sbtmvp_enabled_flag && ph_temporal_mvp_enabled_flag). In an example, the default number N is 5, when at least one of the first flag sps_sbtmvp_enabled_flag and the second flag ph_temporal_mvp_enabled_flag is 0 (SbTMVP is disabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be inferred to be 5. However, when both of the first flag sps_sbtmvp_enabled_flag and the second flag ph_temporal_mvp_enabled_flag are 1 (SbTMVP is enabled), the value of the parameter five_minus_max_num_subblock_merge_cand can be inferred to be 4.

[0162] FIG. 18 shows a flow chart outlining a process (1800) according to an embodiment of the disclosure. The process (1800) can be used in the reconstruction of a block, so to generate a prediction block for the block under reconstruction. In various embodiments, the process (1800) are executed by processing circuitry, such as the processing circuitry in the terminal devices (210), (220), (230) and (240), the processing circuitry that performs functions of the video encoder (303), the processing circuitry that performs functions of the video decoder (310), the processing circuitry that performs functions of the video decoder (410), the processing circuitry that performs functions of the video encoder (503), and the like. In some embodiments, the process (1800) is implemented in software instructions, thus when the processing circuitry executes the software instructions, the processing circuitry performs the process (1800). The process starts at (S1801) and proceeds to (S1810)

[0163] At (S1810), a parameter (e.g., five_minus_max_num_subblock_merge_cand that is indicative of a maximum number of candidates in subblock based merge candidate lists) is

determined based on prediction information decoded from a coded video bitstream. The parameter is in a range that depends on a flag indicative of an enable/disable status of subblock based temporal motion vector prediction. In some examples, an upper limit of the range depends on the flag indicative of the enable/disable status of subblock based temporal motion vector prediction. In an example, the flag is indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level

[0164] In an embodiment, the value of the parameter is signaled in the coded video bitstream. In another example, when the value of the parameter is not signaled in the coded video bitstream, the value of the parameter can be inferred to be the upper limit of the range. For example, the parameter can be inferred based on a default number and the flag indicative of the enable/disable status of subblock based temporal motion vector prediction in response to the parameter not being signaled in the coded video bitstream.

[0165] In some embodiments, the parameter is in a range that depends on a first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level and a second flag indicative of the enable/disable status of temporal motion vector prediction at a picture header (PH) level. In some examples, in response to the parameter not being signaled in the coded video bitstream, the parameter can be inferred based on a default number, the first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at the SPS level and the second flag indicative of the enable/disable status of temporal motion vector prediction at the PH level.

[0166] At (S1820), the maximum number of candidates in the subblock based merge candidate lists is calculated based on the parameter. In some examples, the maximum number of candidates in the subblock based merge candidate lists is calculated by subtracting the parameter from a default number, such as using (Eq. 9). In an example, the default number is 5.

[0167] At (S1830), in response to a current block in a subblock based prediction mode, samples of the current block are reconstructed based on a candidate selection from a constructed subblock based merge candidate list of the current block. The constructed subblock based merge candidate list of the current block is constrained by the maximum number of candidates in the subblock based merge candidate lists.

[0168] Then the process proceeds to (S1899) and terminates.

[0169] 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. 19 shows a computer system (1900) suitable for implementing certain embodiments of the disclosed subject matter.

[0170] 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.

[0171] 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.

[0172] The components shown in FIG. 19 for computer system (1900) 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 (1900).

[0173] Computer system (1900) 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).

[0174] Input human interface devices may include one or more of (only one of each depicted): keyboard (1901), mouse (1902), trackpad (1903), touch screen (1910), data-glove (not shown), joystick (1905), microphone (1906), scanner (1907), camera (1908).

[0175] Computer system (1900) 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 (1910), data-glove (not shown), or joystick (1905), but there can also be tactile feedback devices

that do not serve as input devices), audio output devices (such as: speakers (1909), headphones (not depicted)), visual output devices (such as screens (1910) 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).

[0176] Computer system (1900) can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (1920) with CD/DVD or the like media (1921), thumb-drive (1922), removable hard drive or solid state drive (1923), 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.

[0177] 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.

[0178] Computer system (1900) 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 (1949) (such as, for example USB ports of the computer system (1900)); others are commonly integrated into the core of the computer system (1900) 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 (1900) 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.

[0179] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (1940) of the computer system (1900).

[0180] The core (1940) can include one or more Central Processing Units (CPU) (1941), Graphics Processing Units (GPU) (1942), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (1943), hardware accelerators for certain tasks (1944), and so forth. These devices, along with Read-only memory (ROM) (1945), Random-access memory (1946), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (1947), may be connected through a system bus (1948). In some computer systems, the system bus (1948) 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 (1948), or through a peripheral bus (1949). Architectures for a peripheral bus include PCI, USB, and the like.

[0181] CPUs (1941), GPUs (1942), FPGAs (1943), and accelerators (1944) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (1945) or RAM (1946). Transitional data can be also be stored in RAM (1946), whereas permanent data can be stored for example, in the internal mass storage (1947). 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 (1941), GPU (1942), mass storage (1947), ROM (1945), RAM (1946), and the like.

[0182] 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.

[0183] As an example and not by way of limitation, the computer system having architecture (1900), and specifically the core (1940) 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 (1940) that are of non-transitory nature, such as core-internal mass storage (1947) or ROM (1945). The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core (1940). A computer-readable medium can include one or more memory devices or chips, according to particular needs. The

software can cause the core (1940) 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 (1946) 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 (1944)), 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

[0184] 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.

WHAT IS CLAIMED IS:

1. A method for video decoding in a decoder, comprising:

determining, by a processor, a parameter based on prediction information decoded from a coded video bitstream, the parameter being in a range that depends on a flag indicative of an enable/disable status of subblock based temporal motion vector prediction;

calculating, by the processor, a maximum number of candidates in the subblock based merge candidate lists based on the parameter; and

reconstructing, in response to a current block in a subblock based prediction mode, samples of the current block based on a candidate selection from a constructed subblock based merge candidate list of the current block, the constructed subblock based merge candidate list of the current block being constrained by the maximum number of candidates in the subblock based merge candidate lists.

2. The method of claim 1, further comprising:

calculating the maximum number of candidates in the subblock based merge candidate lists by subtracting the parameter from a default number.

3. The method of claim 2, wherein the default number is 5.

4. The method of claim 1, wherein an upper limit of the range depends on the flag indicative of the enable/disable status of subblock based temporal motion vector prediction.

5. The method of claim 1, further comprising:

receiving the parameter that is signaled in the coded video bitstream.

6. The method of claim 1, further comprising:

inferring the parameter based on a default number and the flag indicative of the enable/disable status of subblock based temporal motion vector prediction in response to the parameter not being signaled in the coded video bitstream.

7. The method of claim 1, wherein the flag is indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level.

8. The method of claim 1, wherein the parameter is in a range that depends on a first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level and a second flag indicative of the enable/disable status of temporal motion vector prediction at a picture header (PH) level.

9. The method of claim 8, further comprising:

inferring the parameter based on a default number, the first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at the SPS level and the second flag indicative of the enable/disable status of temporal motion vector prediction at the PH level in response to the parameter not being signaled in the coded video bitstream.

10. An apparatus for video decoding, comprising:

processing circuitry configured to:

determine a parameter based on prediction information decoded from a coded video bitstream, the parameter being in a range that depends on a flag indicative of an enable/disable status of subblock based temporal motion vector prediction;

calculate a maximum number of candidates in the subblock based merge candidate lists based on the parameter; and

reconstruct, in response to a current block in a subblock based prediction mode, samples of the current block based on a candidate selection from a constructed subblock based merge candidate list of the current block, the constructed subblock based merge candidate list of the current block being constrained by the maximum number of candidates in the subblock based merge candidate lists.

11. The apparatus of claim 10, wherein the processing circuitry is further configured to:

calculate the maximum number of candidates in the subblock based merge candidate lists by subtracting the parameter from a default number.

12. The apparatus of claim 11, wherein the default number is 5.

13. The apparatus of claim 10, wherein an upper limit of the range depends on the flag indicative of the enable/disable status of subblock based temporal motion vector prediction.

14. The apparatus of claim 10, wherein the processing circuitry is configured to

receive the parameter that is signaled in the coded video bitstream.

15. The apparatus of claim 10, wherein the processing circuitry is configured to:

infer the parameter based on a default number and the flag indicative of the enable/disable status of subblock based temporal motion vector prediction in response to the parameter not being signaled in the coded video bitstream.

16. The apparatus of claim 10, wherein the flag is indicative of the enable/disable status of subblock based temporal motion vector prediction at a sequence parameter set (SPS) level.

17. The apparatus of claim 10, wherein the parameter is in a range that depends on a first flag indicative of the enable/disable status of subblock based temporal motion vector prediction

at a sequence parameter set (SPS) level and a second flag indicative of the enable/disable status of temporal motion vector prediction at a picture header (PH) level.

18. The apparatus of claim 17, wherein the processing circuitry is configured to:

infer the parameter based on a default number, the first flag indicative of the enable/disable status of subblock based temporal motion vector prediction at the SPS level and the second flag indicative of the enable/disable status of temporal motion vector prediction at the PH level in response to the parameter not being signaled in the coded video bitstream.

19. A non-transitory computer-readable medium storing instructions which when executed by a computer for video decoding cause the computer to perform:

determining a parameter based on prediction information decoded from a coded video bitstream, the parameter being in a range that depends on a flag indicative of an enable/disable status of subblock based temporal motion vector prediction;

calculating a maximum number of candidates in the subblock based merge candidate lists based on the parameter; and

reconstructing, in response to a current block in a subblock based prediction mode, samples of the current block based on a candidate selection from a constructed subblock based merge candidate list of the current block, the constructed subblock based merge candidate list of the current block being constrained by the maximum number of candidates in the subblock based merge candidate lists.

20. The non-transitory computer-readable medium of claim 19, wherein an upper limit of the range depends on the flag indicative of the enable/disable status of subblock based temporal motion vector prediction.