(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 July 2015 (16.07.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/106078 A1

(51) International Patent Classification:
H04N 19/593 (2014.01) *H04N 19/186* (2014.01)

(21) International Application Number:
PCT/US2015/010783

(22) International Filing Date:
9 January 2015 (09.01.2015)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/925,647	9 January 2014 (09.01.2014)	US
61/926,224	10 January 2014 (10.01.2014)	US
14/592,819	8 January 2015 (08.01.2015)	US

(71) Applicant: **QUALCOMM INCORPORATED** [US/US]; ATTN: International IP Administration, 5775 Morehouse Drive, San Diego, California 92121-1714 (US).

(72) Inventors: **CHEN, Ying**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **PANG, Chao**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **ZHANG, Li**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **SOLE ROJALS, Joel**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **KARCZEWICZ, Marta**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US).

(74) Agent: **JOSEPH, Jeffrey R.**; Shumaker & Sieffert, P.A., 1625 Radio Drive, Suite 300, Woodbury, Minnesota 55125 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH,

[Continued on next page]

(54) Title: INTRA PREDICTION FROM A PREDICTIVE BLOCK

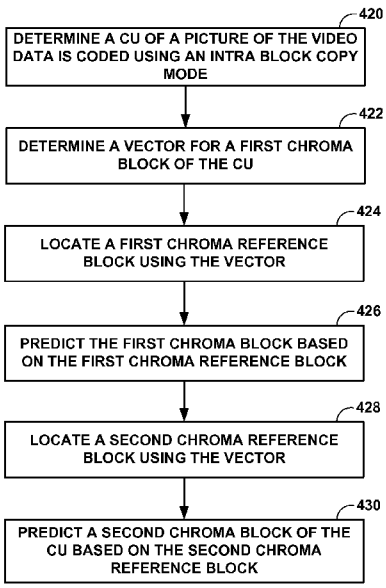

FIG. 9

(57) Abstract: A device for coding video data, the device comprising a memory configured to store video data and a video coder comprising one or more processors configured to: determine a coding unit of a picture of the video data is coded using an intra block copy mode; determine a vector for a first chroma block of the coding unit; locate a first chroma reference block using the vector, wherein the first chroma reference block is in the picture; predict the first chroma block based on the first chroma reference block; locate a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; and predict a second chroma block of the coding unit based on the second chroma reference block.

WO 2015/106078 A1

GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

INTRA PREDICTION FROM A PREDICTIVE BLOCK

[0001] This application claims the benefit of each of the following U.S. Provisional Applications, the entire contents of each of which is incorporated herein by reference:

U.S. Provisional Application Serial No. 61/925,647, filed 9 January 2014;

U.S. Provisional Application Serial No. 61/926,224, filed 10 January 2014.

TECHNICAL FIELD

[0002] This disclosure relates to video coding and, more particularly, prediction of video blocks based on other video blocks.

BACKGROUND

[0003] Digital video capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, tablet computers, e-book readers, digital cameras, digital recording devices, digital media players, video gaming devices, video game consoles, cellular or satellite radio telephones, so-called "smart phones," video teleconferencing devices, video streaming devices, and the like. Digital video devices implement video compression techniques, such as those described in the standards defined by MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), the High Efficiency Video Coding (HEVC) standard presently under development, and extensions of such standards. The video devices may transmit, receive, encode, decode, and/or store digital video information more efficiently by implementing such video compression techniques.

[0004] Video compression techniques perform spatial (intra-picture) prediction and/or temporal (inter-picture) prediction to reduce or remove redundancy inherent in video sequences. For block-based video coding, a video slice (i.e., a video frame or a portion of a video frame) may be partitioned into video blocks, which may also be referred to as treeblocks, coding units (CUs) and/or coding nodes. Video blocks in an intra-coded (I) slice of a picture are encoded using spatial prediction with respect to reference samples in neighboring blocks in the same picture. Video blocks in an inter-coded (P or B) slice of a picture may use spatial prediction with respect to reference samples in neighboring

blocks in the same picture or temporal prediction with respect to reference samples in other reference pictures. Pictures may be referred to as frames, and reference pictures may be referred to a reference frames.

[0005] Spatial or temporal prediction results in a predictive block for a block to be coded. Residual data represents pixel differences between the original block to be coded and the predictive block. An inter-coded block is encoded according to a motion vector that points to a block of reference samples forming the predictive block, and the residual data indicating the difference between the coded block and the predictive block. An intra-coded block is encoded according to an intra-coding mode and the residual data. For further compression, the residual data may be transformed from the pixel domain to a transform domain, resulting in residual transform coefficients, which then may be quantized. The quantized transform coefficients, initially arranged in a two-dimensional array, may be scanned in order to produce a one-dimensional vector of transform coefficients, and entropy coding may be applied to achieve even more compression.

SUMMARY

[0006] In general, this disclosure describes techniques for coding video data that include a mode for intra prediction of blocks of video data from predictive blocks of video data within the same picture, which may be referred to as an IntraBC or IntraMC mode.

[0007] In one example, a method of decoding video data includes determining a coding unit of a picture of the video data is coded using an intra block copy mode; determining from a plurality of vectors associated with the coding unit a vector for a first chroma block of the coding unit; locating a first chroma reference block using the vector, wherein the first chroma reference block is in the picture; predicting the first chroma block based on the first chroma reference block; locating a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; and predicting a second chroma block of the coding unit based on the second chroma reference block.

[0008] In another example, a method of encoding video data includes determining from a plurality of vectors associated with the coding unit a vector for a first chroma block of a coding unit of a picture; locating a first chroma reference block using the vector, wherein the first chroma reference block is in the picture; predicting the first chroma

block based on the first chroma reference block; locating a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; predicting a second chroma block of the coding unit based on the second chroma reference block; and generating, for inclusion in the encoded video data, an indication that the coding unit of the picture is coded using an intra block copy mode and an indication of a vector for each luma block of the coding unit.

[0009] In another example, an apparatus for coding video data includes means for determining from a plurality of vectors associated with the coding unit a coding unit of a picture of the video data is coded using an intra block copy mode; means for determining a vector for a first chroma block of the coding unit; means for locating a first chroma reference block using the vector, wherein the first chroma reference block is in the picture; means for predicting the first chroma block based on the first chroma reference block; means for locating a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; and means for predicting a second chroma block of the coding unit based on the second chroma reference block.

[0010] A device for coding video data, the device comprising a memory configured to store video data and a video coder comprising one or more processors configured to: determine from a plurality of vectors associated with the coding unit a coding unit of a picture of the video data is coded using an intra block copy mode; determine a vector for a first chroma block of the coding unit; locate a first chroma reference block using the vector, wherein the first chroma reference block is in the picture; predict the first chroma block based on the first chroma reference block; locate a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; and predict a second chroma block of the coding unit based on the second chroma reference block.

[0011] In another example, a computer readable storage medium storing instructions that when executed by one or more processors cause the one or more processors to: determine from a plurality of vectors associated with the coding unit a coding unit of a picture of the video data is coded using an intra block copy mode; determine a vector for a first chroma block of the coding unit; locate a first chroma reference block using the vector, wherein the first chroma reference block is in the picture; predict the first chroma block based on the first chroma reference block; locate a second chroma reference block using the vector, wherein the second chroma reference block is in the

picture; and predict a second chroma block of the coding unit based on the second chroma reference block.

[0012] The details of one or more examples are set forth in the accompanying drawings and the description below. Other features, objects, and advantages will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

[0013] FIG. 1 is a block diagram illustrating an example video encoding and decoding system that may utilize the techniques described in this disclosure.

[0014] FIG. 2 is a conceptual diagram illustrating an example predictive block of video data within a current picture for predicting a current block of video data within the current picture according to the techniques of this disclosure.

[0015] FIG. 3 is a conceptual diagram illustrating an example technique for defining an intended region, from which a predictive block of video data may be selected.

[0016] FIG. 4 is a conceptual diagram an example of a current block of video data and neighboring blocks of video data from which candidate predictive vectors for the current block of video data may be derived.

[0017] FIG. 5A shows an example of spatial neighboring motion vector candidates for merge mode.

[0018] FIG. 5B shows an example of spatial neighboring motion vector candidates for Advanced Motion Vector Prediction (AMVP) mode.

[0019] FIG. 6 is a block diagram illustrating an example video encoder that may implement the techniques described in this disclosure.

[0020] FIG. 7 is a block diagram illustrating an example video decoder that may implement the techniques described in this disclosure.

[0021] FIG. 8 is a flow diagram illustrating an example method for encoding video data according to the techniques of this disclosure.

[0022] FIG. 9 is a flow diagram illustrating an example method for decoding video data data according to the techniques of this disclosure.

DETAILED DESCRIPTION

[0023] A video sequence is generally represented as a sequence of pictures. Typically, block-based coding techniques are used to code each of the individual pictures. That is,

each picture is divided into blocks, and each of the blocks is individually coded. Coding a block of video data generally involves forming a predicted value for the block and coding a residual value, that is, the difference between the original block and the predicted value. Specifically, the original block of video data includes a matrix of pixel values, and the predicted value includes a matrix of predicted pixel values. The residual value corresponds to pixel-by-pixel differences between the pixel values of the original block and the predicted pixel values.

[0024] Prediction techniques for a block of video data are generally categorized as intra-prediction or inter-prediction. Intra-prediction, or spatial prediction, generally involves predicting the block from neighboring pixel values that are part of previously coded blocks. Inter-prediction, or temporal prediction, generally involves predicting the block from pixel values of previously coded pictures (e.g., frames or slices).

[0025] Many applications, such as remote desktop, remote gaming, wireless displays, automotive infotainment, cloud computing, and others, are becoming routine in daily lives. Video contents in these applications are usually combinations of natural content, text, artificial graphics, and other contents. In text and artificial graphics regions, repeated patterns (such as characters, icons, symbols, or the like) often exist.

[0026] Intra prediction of blocks of video data from predictive blocks of video data within the same picture, which may be referred to as Intra Block Copy (IntraBC) or Intra Motion Compensation (IntraMC), is a technique which may enable a video coder to remove such redundancy, and improve intra-frame coding efficiency. In some video coding techniques, video coders may use blocks of previously reconstructed video data that are either directly above or below or directly in line horizontally with the current block of video data in the same picture for prediction of the current video block. In other words, if a picture or frame of video data is imposed on a 2-D grid, each block of video data would occupy a unique range of x-values and y-values. Accordingly, some video coders may predict a current block of video data based on blocks of previously coded video data in the same picture that share only the same set of x-values (i.e., vertically in-line with the current video block) or the same set of y-values (i.e., horizontally in-line with the current video block).

[0027] It may be advantageous for a video coder to predict a current video block from a previously reconstructed block of video data within the same frame, i.e., same picture, that is not necessarily directly above or left (or directly right or below), the current block of video data. By including more video blocks in the predictive set, a video coder

may achieve more accurate prediction of the current video block, thereby increasing coding efficiency.

[0028] In general, this disclosure describes techniques for coding video data that include a mode for intra prediction of blocks of video data from predictive blocks of video data within the same picture, which may be referred to as an IntraBC or IntraMC mode. The IntraBC or IntraMC techniques of this disclosure may include identifying a predictive block of video data for the current block of video data. The predictive block of video data may, for example, correspond to a reconstructed block of video data within the same picture as the current block of video data. The predictive block of video data may be from within an intended region that is within the picture, e.g., a region above, above-right, above-left, and/or left of the current block of video data. The predictive block of video data is not limited to being either immediately above or immediately to the left of the current video block and, consequently, the vector used to identify the predictive block relative to the current block is not necessarily a one-dimensional vector. Instead, to identify or determine the predictive block of video data, a video coder may code one or more syntax elements that define a two-dimensional vector that includes a horizontal displacement component and a vertical displacement component relative to the current block of video data. The two-dimensional vector may be referred to as a block vector, offset vector, or motion vector and may, for example, be used to identify a predictive block relative to a top-left corner of the current block.

[0029] As will be explained in greater detail below, blocks of video data referred to as coding units may include multiple prediction units. Each prediction unit may be separately predicted. Thus, for each prediction unit a block vector may be signaled in the bitstream, and a video decoder may use that block vector to predict a luma component of the prediction unit and two chroma components of the prediction unit. Depending on the chroma subsampling format of the video data, the video decoder may scale the block vector in order to predict the two chroma components of the prediction unit. The scaling of the block vector for predicting chroma components is not repeated multiple times to assist with readability. It should be understood that such scaling of the block vector for predicting chroma components may occur when predicting chroma components, but that such scaling of the block vector may not be necessary in every instance of predicting chroma components.

[0030] According to existing techniques, a different block vector may be used to identify reference blocks for each prediction unit of a coding unit. For example, a first

block vector for a first prediction unit of the coding unit may be used to identify a luma reference block for the luma components of the first prediction unit (i.e., prediction block of luma samples), and used to identify chroma reference blocks for the two chroma components of the first prediction unit (i.e., the two corresponding chroma blocks of chroma samples). A second block vector for a second prediction unit of the coding unit may be used to identify a luma reference block for the luma components of the second prediction unit (i.e., prediction block of luma samples), and used to identify chroma reference blocks for the two chroma components of the second prediction unit (i.e., the two corresponding chroma blocks of chroma samples), and so forth.

[0031] According to techniques of this disclosure, however, different block vectors may be used to predict the luma components of each prediction unit of a coding unit, while the same block vector may be used to predict two or more chroma components of prediction units of the coding unit. For example, as above, a first block vector for a first prediction unit of the coding unit may be used to identify a luma reference block for the luma components of the first prediction unit (i.e., prediction block of luma samples), and used to identify chroma reference blocks for the two chroma components of the first prediction unit (i.e., the two corresponding chroma blocks of chroma samples). However, for the two chroma components of the second prediction unit, rather than using the second block vector (as in the above example), the same first block vector is used to identify chroma reference blocks. The second block vector may still be used to identify a luma reference block for the luma components of the second prediction unit.

[0032] More specifically, this disclosure describes techniques to support coding of blocks with blocks predicted within a picture or slice. The proposed techniques may, for example, be used in conjunction with the emerging HEVC range extensions (RCEx), which may support high bit depth (e.g. more than 8 bit) and high chroma sampling format (e.g , including 4:4:4 and 4:2:2). The techniques of this disclosure may also be applied to screen content coding.

[0033] As used herein, the term "video coder" refers generically to both video encoders and video decoders. In this disclosure, the terms "video coding" or "coding" may refer generically to video encoding or video decoding. Thus, unless stated otherwise, it should be assumed that techniques described with reference to coding may be performed by either a video encoder or a video decoder. In some portions of this application, certain techniques may be described with reference to video decoding or to a video decoder. It should not be assumed, however, that such techniques are not

applicable to video encoding or may be not be performed by a video encoder. Such techniques may, for example, be performed as part of determining how to encode video data or may be performed as part of a video decoding loop in a video encoder.

[0034] As used in this disclosure, the term current block refers to a block currently being coded, as opposed to a block that is already coded or yet to be coded. Similarly, a current coding unit, prediction unit, or transform unit, refers to a coding unit, prediction unit, or transform unit that is currently being coded.

[0035] FIG. 1 is a block diagram illustrating an example video encoding and decoding system that may implement one or more of the techniques described herein. As shown in FIG. 1, system 10 includes a source device 12 that provides encoded video data to be decoded by a destination device 14. In particular, source device 12 provides the video data to destination device 14 via a computer-readable medium 16. Source device 12 and destination device 14 may comprise any of a wide range of devices, including desktop computers, notebook (i.e., laptop) computers, tablet computers, set-top boxes, telephone handsets such as so-called "smart" phones, so-called "smart" pads, televisions, cameras, display devices, digital media players, video gaming consoles, video streaming device, or the like. In some cases, source device 12 and destination device 14 may be equipped for wireless communication.

[0036] Destination device 14 may receive the encoded video data to be decoded via computer-readable medium 16. Computer-readable medium 16 may comprise any type of medium or device capable of moving the encoded video data from source device 12 to destination device 14. In one example, computer-readable medium 16 may comprise a communication medium to enable source device 12 to transmit encoded video data directly to destination device 14 in real-time. The encoded video data may be modulated according to a communication standard, such as a wireless communication protocol, and transmitted to destination device 14. The communication medium may comprise any wireless or wired communication medium, such as a radio frequency (RF) spectrum or one or more physical transmission lines. The communication medium may form part of a packet-based network, such as a local area network, a wide-area network, or a global network such as the Internet. The communication medium may include routers, switches, base stations, or any other equipment that may be useful to facilitate communication from source device 12 to destination device 14.

[0037] In some examples, computer-readable medium 16 may comprise a storage device, and source device may output encoded video data via output interface 22 to the

storage device. Similarly, encoded data may be accessed from the storage device by input interface 28. The storage device may include any of a variety of distributed or locally accessed data storage media such as a hard drive, Blu-ray discs, DVDs, CD-ROMs, flash memory, volatile or non-volatile memory, or any other suitable digital storage media for storing encoded video data. For example, a computing device of a medium production facility, such as a disc stamping facility, may receive encoded video data from source device 12 and produce a disc containing the encoded video data. Therefore, computer-readable medium 16 may be understood to include one or more computer-readable media of various forms, in various examples. In a further example, the storage device may correspond to a file server or another intermediate storage device that may store the encoded video generated by source device 12. Destination device 14 may access stored video data from the storage device via streaming or download. The file server may be any type of server capable of storing encoded video data and transmitting that encoded video data to the destination device 14. Example file servers include a web server (e.g., for a website), an FTP server, network attached storage (NAS) devices, or a local disk drive. Destination device 14 may access the encoded video data through any standard data connection, including an Internet connection. This may include a wireless channel (e.g., a Wi-Fi connection), a wired connection (e.g., DSL, cable modem, etc.), or a combination of both that is suitable for accessing encoded video data stored on a file server. The transmission of encoded video data from the storage device may be a streaming transmission, a download transmission, or a combination thereof.

[0038] The techniques of this disclosure are not necessarily limited to wireless applications or settings. The techniques may be applied to video coding in support of any of a variety of multimedia applications, such as over-the-air television broadcasts, cable television transmissions, satellite television transmissions, Internet streaming video transmissions, such as dynamic adaptive streaming over HTTP (DASH), digital video that is encoded onto a data storage medium, decoding of digital video stored on a data storage medium, or other applications. In some examples, system 10 may be configured to support one-way or two-way video transmission to support applications such as video streaming, video playback, video broadcasting, and/or video telephony.

[0039] FIG. 1 is merely an example and the techniques of this disclosure may apply to video coding settings (e.g., video encoding or video decoding) that do not necessarily include any data communication between the encoding and decoding devices. In other

examples, data is retrieved from a local memory, streamed over a network, or the like. A video encoding device may encode and store data to memory, and/or a video decoding device may retrieve and decode data from memory. In many examples, the encoding and decoding is performed by devices that do not communicate with one another, but simply encode data to memory and/or retrieve and decode data from memory.

[0040] In the example of FIG. 1, source device 12 includes video source 18, video encoder 20, and output interface 22. Destination device 14 includes input interface 28, video decoder 30, and display device 32. In other examples, a source device and a destination device may include other components or arrangements. For example, source device 12 may receive video data from an external video source 18, such as an external camera. Likewise, destination device 14 may interface with an external display device 32, rather than including an integrated display device 32.

[0041] The illustrated system 10 of FIG. 1 is merely one example. Any digital video encoding and/or decoding device may perform the techniques described herein. Although generally the techniques of this disclosure are performed by a video encoding device, the techniques may also be performed by a video encoder/decoder, typically referred to as a "CODEC." Moreover, the techniques of this disclosure may also be performed by a video preprocessor. Source device 12 and destination device 14 are merely examples of such coding devices in which source device 12 generates coded video data for transmission to destination device 14. In some examples, devices 12, 14 may operate in a substantially symmetrical manner such that each of devices 12, 14 include video encoding and decoding components. Hence, system 10 may support one-way or two-way video transmission between video devices 12, 14, e.g., for video streaming, video playback, video broadcasting, or video telephony.

[0042] Video source 18 of source device 12 may include a video capture device, such as a video camera, a video archive containing previously captured video, and/or a video feed interface to receive video from a video content provider. As a further alternative, video source 18 may generate computer graphics-based data as the source video, or a combination of live video, archived video, and computer-generated video. In some cases, if video source 18 is a video camera, source device 12 and destination device 14 may form so-called camera phones or video phones. As mentioned above, however, the techniques described in this disclosure may be applicable to video coding in general, and may be applied to wireless and/or wired applications. In each case, the captured, pre-captured, or computer-generated video may be encoded by video encoder 20. The

encoded video information may then be output by output interface 22 onto a computer-readable medium 16. In some examples, output interface 22 may include a modulator/demodulator (modem) and/or a transmitter.

[0043] Input interface 28 of destination device 14 receives information from computer-readable medium 16. In some examples, input interface 28 includes a receiver and/or a modem. The information of computer-readable medium 16 may include an encoded video bitstream, which may include encoded video data and other syntax information defined by video encoder 20, which is used by video decoder 30, and that describes characteristics and/or processing of blocks and other coded units, e.g., slices, pictures, groups of pictures (GOPs), or video data. Display device 32 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a cathode ray tube (CRT), a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device.

[0044] Video encoder 20 and video decoder 30 each may be implemented as any of a variety of suitable encoder circuitry, such as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic, software, hardware, firmware or any combinations thereof. When the techniques are implemented partially in software, a device may store instructions for the software in a suitable, non-transitory computer-readable medium and execute the instructions in hardware using one or more processors to perform the techniques of this disclosure. Each of video encoder 20 and video decoder 30 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined encoder/decoder (CODEC) in a respective device. Although not shown in FIG. 1, in some aspects, video encoder 20 and video decoder 30 may each be integrated with an audio encoder and decoder, and may include appropriate MUX-DEMUX units, or other hardware and software, to handle encoding of both audio and video in a common data stream or separate data streams. If applicable, MUX-DEMUX units may conform to the ITU H.223 multiplexer protocol, or other protocols such as the user datagram protocol (UDP).

[0045] This disclosure may generally refer to video encoder 20 "signaling" certain information to another device, such as video decoder 30. The term "signaling" may generally refer to the communication of syntax elements and/or other data used to decode the compressed video data. Such communication may occur in real- or near-real-time. Alternately, such communication may occur over a span of time, such as

might occur when storing syntax elements to a computer-readable storage medium in an encoded bitstream at the time of encoding, which then may be retrieved by a decoding device at any time after being stored to this medium.

[0046] Video encoder 20 and video decoder 30 are described in this disclosure, for purposes of illustration, as being configured to operate according to one or more video coding standards. However, the techniques of this disclosure are not necessarily limited to any particular coding standard, and may be applied for a variety of different coding standards.

[0047] Video encoder 20 and video decoder 30 may operate according to a video coding standard, such as the High Efficiency Video Coding (HEVC) standard presently under development, including any extensions, modifications, or additions, and may conform to the HEVC Test Model (HM). Alternatively, video encoder 20 and video decoder 30 may operate according to other proprietary or industry standards, such as the ITU-T H.264 standard, alternatively referred to as MPEG-4, Part 10, Advanced Video Coding (AVC), or extensions of such standards. The techniques of this disclosure, however, are not limited to any particular coding standard. Other examples of video coding standards include MPEG-2 and ITU-T H.263.

[0048] Examples of other proprietary or industry standards include the ITU-T H.261, ISO/IEC MPEG-1 Visual, ITU-T H.262 or ISO/IEC MPEG-2 Visual, ITU-T H.263, ISO/IEC MPEG-4 Visual and ITU-T H.264 (also known as ISO/IEC MPEG-4 AVC), including its Scalable Video Coding (SVC) and Multiview Video Coding (MVC) extensions, or extensions of, modifications of, or additions to, such standards. Furthermore, there is an ongoing effort to generate a three-dimensional video (3DV) coding extension to H.264/AVC, namely AVC-based 3DV. In other words, example video coding standards include ITU-T H.261, ISO/IEC MPEG-1 Visual, ITU-T H.262 or ISO/IEC MPEG-2 Visual, ITU-T H.263, ISO/IEC MPEG-4 Visual and ITU-T H.264 (also known as ISO/IEC MPEG-4 AVC), including its Scalable Video Coding (SVC) and Multi-view Video Coding (MVC) extensions. The H.264 standard is described in ITU-T Recommendation H.264, Advanced Video Coding for generic audiovisual services, by the ITU-T Study Group, and dated March, 2005, which may be referred to herein as the H.264 standard or H.264 specification, or the H.264/AVC standard or specification. The Joint Video Team (JVT) continues to work on extensions to H.264/MPEG-4 AVC. A recent joint draft of MVC is described in "Advanced video coding for generic audiovisual services," ITU-T Recommendation H.264, Mar. 2010.

[0049] In addition, there is a new developed video coding standard, namely High Efficiency Video Coding (HEVC), developed by the Joint Collaboration Team on Video Coding (JCT-VC) of ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Motion Picture Experts Group (MPEG). In the example of FIG. 1, video encoder 20 and video decoder 30 may operate according to HEVC. Bross et al., "High Efficiency Video Coding (HEVC) text specification draft 10 (for FDIS & Last Call), Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 12th Meeting: Geneva, CH, 14–23 Jan. 2013, document JCTVC-L1003_v34, which as of October 17, 2013 is available from: http://phenix.int-evry.fr/jct/doc_end_user/documents/12_Geneva/wg11/JCTVC-L1003-v34.zip, hereinafter, "JCTVC-L1003-v34", the entire content of which is incorporated by reference, is a recent draft of HEVC. The version of HEVC defined in JCTVC-L1003-v34 may be referred to as HEVC version 1 or "HEVC v1." McCann et al., "High Efficiency Video Coding (HEVC) Test Model 12 (HM 12) Encoder Description," document JCTVC-N1002, available from http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=8143, the entire content of which is incorporated by reference, is a recent encoder description of HEVC.

[0050] Two extensions of HEVC supporting 3D services have been under development by the Joint Collaboration Team on 3D Video coding (JCT-3V) of ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Motion Picture Experts Group (MPEG). The two extensions are namely MV-HEVC and 3D-HEVC respectively. In the example of FIG. 1, video encoder 20 and video decoder 30 may operate according to MV-HEVC and/or 3D-HEVC.

[0051] MV-HEVC supports the coding of multiple (texture) views without changing block level design. Tech et al., "MV-HEVC Draft Text 5," Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, document no. JCT3V-E1004-v6, available from http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/5_Vienna/wg11/JCT3V-E1004-v6.zip, the entire content of which is incorporated by reference, is a recent draft of MV-HEVC.

[0052] 3D-HEVC codes multi-view video plus depth format and includes new coding tools built in addition to the HEVC coding modules. The newly introduced coding tools are applicable for both texture coding and depth coding. Tech et al., "3D-JEVC Draft Text 1," Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 5th Meeting: Vienna, AT, 27

Jul. –2 Aug. 2013, document no. JCT3V-E1001-v3, available from http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/5_Vienna/wg11/JCT3V-E1001-v3.zip, the entire content of which is incorporated by reference, is a recent draft of 3D-HEVC. A recent software 3D-HTM for 3D-HEVC can be downloaded from the following link: [3D-HTM version 8.0]: https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/HTM-8.0/. A recent software description, Zhang et al., "3D-HEVC Test Model 5," Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 5th Meeting: Vienna, AT, 27 Jul. –2 Aug. 2013, document number: JCT3V-E1005, the entire content of which is incorporated by reference, is available from: http://phenix.it-sudparis.eu/jct2/doc_end_user/current_document.php?id=1360.

[0053] Additional development and extensions of the HEVC standard include the HEVC Range Extensions. An example of the HEVC Range Extensions is described in Flynn et al., "High Efficiency Video Coding (HEVC) Range Extensions text specification: Draft 4," Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Document JCTVC-N1005_v3, 13th Meeting: Incheon, KR, 18–26 April 2013, which is incorporated herein by reference in its entirety, and is available at: http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=8139.

[0054] Additional development and extensions of the HEVC standard include the screen content coding (SCC) extension.

[0055] In HEVC and other video coding specifications, a video sequence typically includes a series of pictures. Pictures may also be referred to as "frames." A picture may include three sample arrays, denoted S_L, S_{Cb} and S_{Cr}. S_L is a two-dimensional array (i.e., a block) of luma samples. S_{Cb} is a two-dimensional array of Cb chrominance samples. S_{Cr} is a two-dimensional array of Cr chrominance samples. Chrominance samples may also be referred to herein as "chroma" samples. In other instances, a picture may be monochrome and may only include an array of luma samples.

[0056] The HEVC standardization efforts are based on an evolving model of a video coding device referred to as the HEVC Test Model (HM). The HM presumes several additional capabilities of video coding devices relative to existing devices according to, e.g., ITU-T H.264/AVC. For example, whereas H.264 provides nine intra-prediction

encoding modes, the HM may provide as many as thirty-three intra-prediction encoding modes.

[0057] In general, the working model of the HM describes that video encoder 20 may divide a video frame or picture into a sequence of coding tree units (CTUs), also referred to as largest coding units (LCUs) or treeblocks. Each of the CTUs may comprise a coding tree block (CTB) of luma samples, two corresponding coding tree blocks of chroma samples, and syntax structures used to code the samples of the coding tree blocks. In a monochrome picture or a picture comprising separate color planes, a CTU may comprise a single coding tree block and syntax structures used to code the samples of the single coding tree block.

[0058] Syntax data within a bitstream may define a size for the CTU, which is a largest coding unit in terms of the number of pixels. A CTU may be an NxN block of samples. The CTUs of HEVC may be broadly analogous to the macroblocks of other video coding standards, such as H.264/AVC. However, a CTU is not necessarily limited to a particular size and may include one or more coding units (CUs).

[0059] A video frame or picture may be partitioned into one or more slices. A slice includes a number of consecutive CTUs in coding or scanning order. Each CTU may be split into coding units (CUs) according to a quadtree. A CU may be an NxN block of samples. In general, a quadtree data structure includes one node per CU, with a root node corresponding to the treeblock. If a CU is split into four sub-CUs, the node corresponding to the CU includes four leaf nodes, each of which corresponds to one of the sub-CUs. A CU or sub-CU, which may also be referred to as a CU, may comprise a coding block of luma samples and two corresponding coding blocks of chroma samples of a picture that has a luma sample array, a Cb sample array, and a Cr sample array, and syntax structures used to code the samples of the coding blocks. In a monochrome picture or a picture comprising separate color planes, a CU may comprise a single coding block and syntax structures used to code the samples of the single coding block.

[0060] Each node of the quadtree data structure may provide syntax data for the corresponding CU. For example, a node in the quadtree may include a split flag, indicating whether the CU corresponding to the node is split into sub-CUs. Syntax elements for a CU may be defined recursively, and may depend on whether the CU is split into sub-CUs. If a CU is not split further, it is referred as a leaf-CU. In this disclosure, four sub-CUs of a leaf-CU may also be referred to as leaf-CUs even if there is no explicit splitting of the original leaf-CU. For example, if a CU at 16x16 size is not

split further, the four 8x8 sub-CUs may also be referred to as leaf-CUs although the 16x16 CU was never split. Syntax data associated with a coded bitstream may define a maximum number of times a treeblock may be split, referred to as a maximum CU depth, and may also define a minimum size of the coding nodes. Accordingly, a bitstream may also define a smallest coding unit (SCU).

[0061] A CU includes a coding node and prediction units (PUs) and transform units (TUs) associated with the coding node. A size of the CU corresponds to a size of the coding node and must be square in shape. The size of the CU may range from 8x8 pixels up to the size of the treeblock with a maximum of 64x64 pixels or greater. In general, video encoder 20 and video decoder 30 code each CU with one mode, e.g., intra prediction or inter prediction.

[0062] Each CU may contain one or more PUs and one or more TUs. Syntax data associated with a CU may describe, for example, partitioning of the CU into one or more PUs. Partitioning modes may differ between whether the CU is skip or direct mode encoded, intra-prediction mode encoded, or inter-prediction mode encoded. PUs may be partitioned to be non-square (e.g., rectangular) in shape. Syntax data associated with a CU may also describe, for example, partitioning of the CU into one or more TUs according to a quadtree. A TU can be square or non-square (e.g., rectangular) in shape.

[0063] PUs and TUs of a CU may comprise a prediction block of luma samples, two corresponding prediction blocks of chroma samples of a picture, and syntax structures used to predict the prediction block samples. Video encoder 20 may generate predictive luma, Cb, and Cr blocks for each PU of the CU. In a monochrome picture or a picture comprising separate color planes, a PU may comprise a single prediction block and syntax structures used to predict the single prediction block.

[0064] Video encoder 20 and video decoder 30 may support PUs having various sizes. Assuming that the size of a particular CU is 2Nx2N, video encoder 20 and video decoder 30 may support intra-prediction in PU sizes of 2Nx2N or NxN, and inter-prediction in symmetric PU sizes of 2Nx2N, 2NxN, Nx2N, or NxN. Video encoder 20 and video decoder 30 may also support asymmetric partitioning for inter-prediction in PU sizes of 2NxnU, 2NxnD, nLx2N, and nRx2N. In asymmetric partitioning, one direction of a CU is not partitioned, while the other direction is partitioned into 25% and 75%. The portion of the CU corresponding to the 25% partition is indicated by an "n" followed by an indication of "Up", "Down," "Left," or "Right." Thus, for example,

"2NxnU" refers to a 2Nx2N CU that is partitioned horizontally with a 2Nx0.5N PU on top and a 2Nx1.5N PU on bottom. In HEVC, the smallest PU sizes are 8x4 and 4x8.

[0065] In this disclosure, "NxN" and "N by N" may be used interchangeably to refer to the pixel dimensions of a video block in terms of vertical and horizontal dimensions, e.g., 16x16 pixels or 16 by 16 pixels. In general, a 16x16 block has 16 pixels in a vertical direction (y = 16) and 16 pixels in a horizontal direction (x = 16). Likewise, an NxN block generally has N pixels in a vertical direction and N pixels in a horizontal direction, where N represents a nonnegative integer value. The pixels in a block may be arranged in rows and columns. Moreover, blocks need not necessarily have the same number of pixels in the horizontal direction as in the vertical direction. For example, blocks may comprise NxM pixels, where M is not necessarily equal to N.

[0066] In general, a PU represents a spatial area corresponding to all or a portion of the corresponding CU, and may include data for retrieving a reference sample for the PU. Moreover, a PU includes data related to prediction. For example, when the PU is intra-mode encoded, data for the PU may describe an intra-prediction mode for a prediction block corresponding to the PU.

[0067] As another example, when the PU is inter-mode encoded, the PU may include data defining one or more motion vectors for the PU. The data defining the motion vector for a PU may describe, for example, a horizontal component of the motion vector, a vertical component of the motion vector, a resolution for the motion vector (e.g., one-quarter pixel precision or one-eighth pixel precision). The PU may also include data identifying a reference picture to which the motion vector points, such as an index into reference picture list that includes the reference picture.

[0068] Inter prediction may be uni-directional (i.e., uni-prediction) or bi-directional (i.e., bi-prediction). To perform uni-prediction or bi-prediction, video encoder 20 andd video decoder may generate a first reference picture list (RefPicList0) and a second reference picture list (RefPicList1) for a current picture. Each of the reference picture lists may include one or more reference pictures. After a reference picture list is constructed (namely RefPicList0 and RefPicList1 if available), a reference index to a reference picture list can be used to identify any reference picture included in the reference picture list.

[0069] A reference picture may be a previous picture in temporal, e.g., display, order, a future picture, or a combination of predictions from two or more previously encoded pictures. Video coders, e.g., video encoder 20 and video decoder 30 use picture order

count (POC) to identify a temporal order of a picture. Video coders also use POC values of pictures for reference picture list construction and motion vector scaling. Video encoder 20 or video decoder 30 may store the reference pictures in a reference picture memory, e.g., reference picture memory 368 (FIG. 6) and reference picture memory 396 (FIG. 7).

[0070] When a current video block, e.g., PU, is coded using IntraBC according to the techniques of this disclosure, data defining a two-dimensional vector (which may be referred to as a block vector, offset vector, motion vector, or displacement vector) for the block may describe, for example, a horizontal component of the vector, a vertical component of the vector, a resolution for the vector (e.g., one-quarter pixel precision or one-eighth pixel precision). However, the data of the PU predicted using IntraBC according to the techniques of this disclosure need not identify a reference picture to which the motion vector points, as the reference block is within same frame or pictures as the current video block.

[0071] After video encoder 20 generates predictive luma, Cb, and Cr blocks for one or more PUs of a CU, video encoder 20 may generate a luma residual block for the CU. Each sample in the CU's luma residual block indicates a difference between a luma sample in a predictive luma block of a PU of the CU and a corresponding sample in the CU's original luma coding block. In addition, video encoder 20 may generate a Cb residual block for the CU. Each sample in the CU's Cb residual block may indicate a difference between a Cb sample in a predictive Cb block of a PU of the CU and a corresponding sample in the CU's original Cb coding block. Video encoder 20 may also generate a Cr residual block for the CU. Each sample in the CU's Cr residual block may indicate a difference between a Cr sample in a predictive Cr block of a PU of the CU and a corresponding sample in the CU's original Cr coding block.

[0072] A leaf-CU having one or more PUs may also include one or more transform units (TUs). A TU of a CU may comprise a transform block of luma samples, two corresponding transform blocks of chroma samples, and syntax structures used to transform the transform block samples. Video encoder 20 may transform pixel difference values within the residual blocks associated with the TUs to produce transform coefficients, which may be quantized. Video encoder 20 may, for examples, apply discrete cosine transform (DCT), an integer transform, a wavelet transform, or a conceptually similar transform to the residual video data.

[0073] The TUs may be specified using an RQT (also referred to as a TU quadtree structure), as discussed above. For example, a split flag may indicate whether a leaf-CU is split into four transform units. Then, each transform unit may be split further into further sub-TUs. When a TU is not split further, it may be referred to as a leaf-TU. TU is not necessarily limited to the size of a PU. Thus, TUs may be larger or smaller than a PU. For intra coding, a PU may be collocated with a corresponding leaf-TU for the same CU. In some examples, the maximum size of a leaf-TU may correspond to the size of the corresponding leaf-CU. In general, this disclosure uses the terms CU and TU to refer to leaf-CU and leaf-TU, respectively, unless noted otherwise.

[0074] This disclosure may use the term "video unit," "video block," or "block of video data" to refer to any one or more blocks of samples and syntax structures used to code samples of the one or more blocks of samples as described herein. Example types of video blocks may include CTUs, CUs, PUs, or TUs in the context of HEVC, or similar data structures in the context of other standards (e.g., macroblocks or macroblock partitions thereof in H.264/AVC).

[0075] Following any transforms to produce transform coefficients, video encoder 20 may perform quantization of the transform coefficients. Quantization generally refers to a process in which transform coefficients are quantized to possibly reduce the amount of data used to represent the coefficients, providing further compression. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, an n-bit value may be rounded down to an m-bit value during quantization, where n is greater than m.

[0076] Following quantization, the video encoder may scan the transform coefficients, producing a one-dimensional vector from the two-dimensional matrix including the quantized transform coefficients. The scan may be designed to place higher energy (and therefore lower frequency) coefficients at the front of the array and to place lower energy (and therefore higher frequency) coefficients at the back of the array. In some examples, video encoder 20 may utilize a predefined scan order to scan the quantized transform coefficients to produce a serialized vector that can be entropy encoded. In other examples, video encoder 20 may perform an adaptive scan. After scanning the quantized transform coefficients to form a one-dimensional vector, video encoder 20 may entropy encode the one-dimensional vector, e.g., according to context-adaptive variable length coding (CAVLC), context-adaptive binary arithmetic coding (CABAC), syntax-based context-adaptive binary arithmetic coding (SBAC), Probability Interval

Partitioning Entropy (PIPE) coding or another entropy encoding methodology. Video encoder 20 may also entropy encode syntax elements associated with the encoded video data for use by video decoder 30 in decoding the video data.

[0077] To perform CABAC, video encoder 20 may assign a context within a context model to a symbol to be transmitted. The context may relate to, for example, whether neighboring values of the symbol are non-zero or not. To perform CAVLC, video encoder 20 may select a variable length code for a symbol to be transmitted. Codewords in VLC may be constructed such that relatively shorter codes correspond to more probable symbols, while longer codes correspond to less probable symbols. In this way, the use of VLC may achieve a bit savings over, for example, using equal-length codewords for each symbol to be transmitted. The probability determination may be based on a context assigned to the symbol.

[0078] In addition, video encoder 20 may decode encoded pictures, e.g., by inverse quantizing and inverse transforming residual data, and combine the residual data with prediction data. In this manner, video encoder 20 can simulate the decoding process performed by video decoder 30. Both video encoder 20 and video decoder 30, therefore, have access to substantially the same decoded video data, e.g., pictures or blocks from pictures, for use in intra-picture, inter-picture, or IntraBC prediction.

[0079] Video encoder 20 may output an encoded video bitstream that includes a sequence of bits that form a representation of the coded pictures and associated data, including syntax elements. The bitstream may comprise a sequence of network abstraction layer (NAL) units. Each of the NAL units may include a NAL unit header and may encapsulate a raw byte sequence payload (RBSP). The NAL unit header may include a syntax element that indicates a NAL unit type code. The NAL unit type code specified by the NAL unit header of a NAL unit indicates the type of the NAL unit. A RBSP may comprise a syntax structure containing an integer number of bytes that is encapsulated within a NAL unit. In some instances, an RBSP includes zero bits.

[0080] Different types of NAL units may encapsulate different types of RBSPs. For example, a first type of NAL unit may encapsulate an RBSP for a parameter set, a second type of NAL unit may encapsulate an RBSP for a coded slice, a third type of NAL unit may encapsulate an RBSP for Supplemental Enhancement Information (SEI), and so on. NAL units that encapsulate RBSPs for video coding data (as opposed to RBSPs for parameter sets and SEI messages) may be referred to as video coding layer (VCL) NAL units. A NAL unit that encapsulates a coded slice may be referred to

herein as a coded slice NAL unit. An RBSP for a coded slice may include a slice header and slice data.

[0081] Video encoder 20 may include in the encoded video bitstream, in addition to the encoded video data, syntax elements that inform video decoder how to decode a particular block of video data, or grouping thereof. Video encoder 20 may include the syntax elements in a variety of syntax structures, e.g., depending on the type of video structure (e.g., sequence, picture, slice, block) to which it refers, and how frequently its value may change. For example, video encoder 20 may include syntax elements in parameter sets, such as a Video Parameter Set (VPS), Sequence Parameter Set (SPS), or Picture Parameter Set (PPS). As other examples, video encoder 20 may include syntax elements in SEI messages and slice headers.

[0082] In general, video decoder 30 may perform a decoding process that is the inverse of the encoding process performed by video encoder. For example, video decoder 30 may perform entropy decoding using the inverse of the entropy encoding techniques used by video encoder to entropy encode the quantized video data. Video decoder 30 may further inverse quantize the video data using the inverse of the quantization techniques employed by video encoder 20, and may perform an inverse of the transformation used by video encoder 20 to produce the transform coefficients that quantized. Video decoder 30 may then apply the resulting residual blocks to adjacent reference video data (intra-prediction), predictive blocks from another picture (inter-prediction), or predictive blocks from the same picture (IntraBC) to produce the video block for eventual display. Video decoder 30 may be configured, instructed, controlled or directed to perform the inverse of the various processes performed by video encoder 20 based on the syntax elements provided by video encoder 20 with the encoded video data in the bitstream received by video decoder 30.

[0083] Each picture may comprise a luma component and one or more chroma components. Accordingly, the block-based encoding and decoding operations described herein may be equally applicable to blocks, e.g., CUs, PUs and TUs, including or associated with luma or chroma pixel values.

[0084] IntraMC (also referred to as IntraBC) is a dedicated technique which enables removing intra-picture redundancy and improving the intra-frame coding efficiency as reported in Budagavi et al., "AHG8: Video coding using Intra motion compensation," Document: JCTVC-M0350, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 13th Meeting: Incheon, KR, 18-

26 Apr. 2013 (hereinafter, "JCT-VC M0350"). JCT-VC M0350 is incorporated herein by reference in its entirety, and is available for download from http://phenix.int-evry.fr/jct/doc_end_user/current_document.php?id=7601. According to JCT-VC M0350, IntraMC includes coding: (1) a one-dimensional offset or displacement vector (also called here block vector, motion vector, or "MV"), which indicates the position of the prediction signal, e.g., a block within the same frame or picture, displaced from the current video block, e.g., CU, together with (2) the residual signal. For CUs or other blocks which use IntraMC, the prediction signals are obtained from the already reconstructed region in the same picture.

[0085] FIG. 2 is a conceptual diagram illustrating an example technique for predicting a current block of video data 102 within a current picture 103 according to a mode for intra prediction of blocks of video data from predictive blocks of video data within the same picture according to this disclosure, e.g., according to an IntraBC mode in accordance with the techniques of this disclosure. FIG. 2 illustrates a predictive block of video data 104 within current picture 103. A video coder, e.g., video encoder 20 and/or video decoder 30, may use predictive video block 104 to predict current video block 102 according to an IntraBC mode in accordance with the techniques of this disclosure.

[0086] Video encoder 20 selects predictive video block 104 for predicting current video block 102 from a set of previously reconstructed blocks of video data. Video encoder 20 reconstructs blocks of video data by inverse quantizing and inverse transforming the video data that is also included in the encoded video bitstream, and summing the resulting residual blocks with the predictive blocks used to predict the reconstructed blocks of video data. In the example of FIG. 2, intended region 108 within picture 103, which may also be referred to as an "intended area" or "raster area," includes the set of previously reconstructed video blocks. Video encoder 20 may define intended region 108 within picture 103 in variety of ways, as described in greater detail below. Video encoder 20 may select predictive video block 104 to predict current video block 102 from among the video blocks in intended region 108 based on an analysis of the relative efficiency and accuracy of predicting and coding current video block 102 based on various video blocks within intended region 108.

[0087] Video encoder 20 determines two-dimensional vector 106 representing the location or displacement of predictive video block 104 relative to current video block 102. Two-dimensional block vector 106 includes horizontal displacement component

112 and vertical displacement component 110, which respectively represent the horizontal and vertical displacement of predictive video block 104 relative to current video block 102. Video encoder 20 may include one or more syntax elements that identify or define two-dimensional block vector 106, e.g., that define horizontal displacement component 112 and vertical displacement component 110, in the encoded video bitstream. Video decoder 30 may decode the one or more syntax elements to determine two-dimensional block vector 106, and use the determined vector to identify predictive video block 104 for current video block 102.

[0088] In some examples, the resolution of two-dimensional block vector 106 can be integer pixel, e.g., be constrained to have integer pixel resolution. In such examples, the resolution of horizontal displacement component 112 and vertical displacement component 110 may be integer pixel. In such examples, video encoder 20 and video decoder 30 need not interpolate pixel values of predictive video block 104 to determine the predictor for current video block 102.

[0089] In other examples, the resolution of one or both of horizontal displacement component 112 and vertical displacement component 110 can be sub-pixel. For example, one of components 112 and 114 may have integer pixel resolution, while the other has sub-pixel resolution. In some examples, the resolution of both of horizontal displacement component 112 and vertical displacement component 110 can be sub-pixel, but horizontal displacement component 112 and vertical displacement component 110 may have different resolutions.

[0090] In some examples, a video coder, e.g., video encoder 20 and/or video decoder 30, adapts the resolution of horizontal displacement component 112 and vertical displacement component 110 based on a specific level, e.g., block-level, slice-level, or picture-level adaptation . For example, video encoder 20 may signal a flag at the slice level, e.g., in a slice header, that indicates whether the resolution of horizontal displacement component 112 and vertical displacement component 110 is integer pixel resolution or is not integer pixel resolution. If the flag indicates that the resolution of horizontal displacement component 112 and vertical displacement component 110 is not integer pixel resolution, video decoder 30 may infer that the resolution is sub-pixel resolution. In some examples, one or more syntax elements, which are not necessarily a flag, may be transmitted for each slice or other unit of video data to indicate the collective or individual resolutions of horizontal displacement components 112 and/or vertical displacement components 110.

[0091] In still other examples, instead of a flag or a syntax element, video encoder 20 may set based on, and video decoder 30 may infer the resolution of horizontal displacement component 112 and/or vertical displacement component 110 from resolution context information. Resolution context information may include, as examples, the color space (e.g., YUV, RGB, or the like), the specific color format (e.g., 4:4:4, 4:2:2, 4:2:0, or the like), the frame size, the frame rate, or the quantization parameter (QP) for the picture or sequence of pictures that include current video block 102. In at least some examples, a video coder may determine the resolution of horizontal displacement component 112 and/or vertical displacement component 110 based on information related to previously coded frames or pictures. In this manner, the resolution of horizontal displacement component 112 and the resolution for vertical displacement component 110 may be pre-defined, signaled, may be inferred from other, side information (e.g., resolution context information), or may be based on already coded frames.

[0092] Current video block 102 may be a CU, or a PU of a CU. In some examples, a video coder, e.g., video encoder 20 and/or video decoder 30, may split a CU that is predicted according to IntraBC into a number of PUs. In such examples, the video coder may determine a respective (e.g., different) two-dimensional vector 106 for each of the PUs of the CU. For example, a video coder may split a 2Nx2N CU into two 2NxN PUs, two Nx2N PUs, or four NxN PUs. As other examples, a video coder may split a 2Nx2N CU into ((N/2)xN + (3N/2)xN) PUs, ((3N/2)xN + (N/2)xN) PUs, (Nx(N/2) + Nx(3N/2)) PUs, (Nx(3N/2) + Nx(N/2)) PUs, four (N/2)x2N PUs, or four 2Nx(N/2) PUs. In some examples, video coder may predict a 2Nx2N CU using a 2Nx2N PU.

[0093] Current video block 102 may be a luma video block, or a chroma video block corresponding to a luma video block. In some examples, video encoder 20 may only encode one or more syntax elements defining two-dimensional vectors 106 for luma video blocks into the encoded video bitstream. In such examples, video decoder 30 may derive two-dimensional vectors 106 for each of one or more chroma blocks corresponding to a luma block based on the two-dimensional vector signaled for the luma block.

[0094] Depending on the color format, e.g., color sampling format or chroma sampling format, a video coder may downsample corresponding chroma video blocks relative to the luma video block. Color format 4:4:4 does not include downsampling, meaning that

the chroma blocks include the same number of samples in the horizontal and vertical directions as the luma block. Color format 4:2:2 is downsampled in the horizontal direction, meaning that there are half as many samples in the horizontal direction in the chroma blocks relative to the luma block. Color format 4:2:0 is downsampled in the horizontal and vertical directions, meaning that there are half as many samples in the horizontal and vertical directions in the chroma blocks relative to the luma block.

[0095] In examples in which video coders determine vectors 106 for chroma video blocks based on vectors 106 for corresponding luma blocks, the video coders may need to modify the luma vector. For example, if a luma vector 106 has integer resolution with horizontal displacement component 112 and/or vertical displacement component 110 being an odd number of pixels, and the color format is 4:2:2 or 4:2:0, the converted luma vector may not point an integer pixel location in the corresponding chroma block. In such examples, video coders may scale the luma vector for use as a chroma vector to predict a corresponding chroma block. In some examples, video encoder 20 may define intended region 108, or scale the converted vector, such that a converted luma vector 106 used for predicting a chroma block may not point to predictive chroma blocks that are not reconstructed, or that are in-loop filtered. In this disclosure, if scaling of the vector for the chroma block is not explicitly mentioned, it should not be assumed that such scaling does not occur. Scaling of the vector for the chroma block may occur (but may not be necessary in every case) even if not explicitly described in the examples described in this disclosure.

[0096] FIG. 3 is a conceptual diagram illustrating an example technique for defining an intended region, from which video encoder 20 may select a predictive block of video data to predict a current block of video data. In the example illustrated by FIG. 3, video encoder 20 is predicting and encoding current block of video data 122. Video encoder 20 selects predictive block of video data 124 within intended region 128 to predict current video block 122. Video encoder 20 determines two-dimensional vector 126, which includes a horizontal displacement component 132 and a vertical displacement component 130, that indicates the displacement of predictive video block 124 relative to current video block 122. Video encoder 20 encodes one or more syntax elements in the encoded video bitstream defining two-dimensional vector 126.

[0097] In some examples, video encoder 20 defines intended region 128, e.g., defines the size of intended region, such as a height, width, or other dimension, such that implementation and processing complexity, particularly at video decoder 30, can be

reduced. In doing so, video encoder 20 may limit the size of two-dimensional vector 126, e.g., limits the size of vertical displacement component 130 and/or horizontal displacement component 132. In some examples, video encoder 20 may limit an intended region 128 to facilitate parallel processing video data by video encoder 20 and video decoder 30. In some examples, video encoder 20 may limit intended region 128 to facilitate use of predictive video blocks without in-loop filtering, such as deblocking and sample adaptive offset (SAO) filtering, and without unduly burdening the memory requirements of a video coder or delaying application of such in-loop filtering.

[0098] As illustrated in FIG. 3, current video block 122 is within a current LCU 134. FIG. 3 also illustrates the left-neighboring (left) LCU 136 of current LCU. Left LCU 136 is illustrated in FIG. 3 because video blocks of a picture are typically encoded in raster-scan order from top-left to bottom-right. In examples in which video blocks of a picture are coded in a different order, the following discussion regarding left LCU 136 may apply to a different, neighboring LCU of current LCU 134.

[0099] In some examples, video encoder 20 may limit intended region 128 such that the fetched predictive video block 124 is within the same LCU as current video block 122, i.e., within current LCU 134. Limiting intended region 128 to current LCU 134 may facilitate parallel processing of LCUs by video coders, because processing units of a video coder may not require information from another LCU when coding a block of a current LCU.

[0100] In some examples in which intended region 128 is limited to current LCU 134, two-dimensional vector 126 may be limited to a horizontal vector if current video block 122 is a top-most block of current LCU 134, and to a vertical vector if current video block 122 is a left-most block of current LCU 134. In such examples, video encoder 20 may encode one or more syntax elements defining horizontal displacement component 132 of two-dimensional vector 126, and need not encode one or more syntax elements defining vertical displacement component 130 of two-dimensional vector 126, which would be zero, if current video block 122 is a top-most block of current LCU 134. Similarly, video encoder 20 may encode one or more syntax elements defining vertical displacement component 130 of two-dimensional vector 126, and need not encode one or more syntax elements defining horizontal displacement component 132 of two-dimensional vector 126, which would be zero, if current video block 122 is a left-most block of current LCU 134. Similarly, if intended region 128 is limited to current LCU 134 and current video block 122 is the top-left unit of current LCU 134, both horizontal

and vertical components 130, 132 of two-dimensional vector 126 have to be zero. In some examples in which this situation is encountered, video encoder 20 may not perform IntraBC, and need not single any syntax elements for IntraBC, such as syntax elements to indicate two-dimensional vector 126, or any flag indicating whether current video block 122 is predicted according to IntraBC.

[0101] These techniques to reduce signaling for IntraBC when an intended region is limited to a current LCU and a current video block is within certain positions in the current LCU may also be applied for IntraBC techniques limited to one-dimensional motion vectors, such as those proposed in JCT-VC M0350 as well. For example, if the vertical motion vector is restricted to be within the current LCU and the current unit is the top-most unit, there is no need to signal whether the motion is vertical or horizontal.

[0102] In some examples, video encoder 20 may limit intended region 128 to be within current LCU 134 and part of the reconstructed area in one or more neighboring LCUs, e.g., left LCU 136 as illustrated in FIG. 3. In this manner, video encoder 20 limits two-dimensional vectors 126 such that they may refer, in addition to reconstructed blocks in current LCU 134, only to blocks in a reconstructed area of one or more neighboring LCUs, such as left LCU 136, as illustrated in FIG. 3.

[0103] In some examples, video encoder 20 limits a size of intended region 128 based on a size of current LCU 134. For example, as illustrated in FIG. 3, video encoder 20 may limit a height 138 of intended region 128 based on a height (or upper vertical limit) of current LCU 134, such that intended region 128 does not extend vertically beyond current LCU 134 and left LCU 136. The size of an LCU (or CTB) may be signaled by video encoder 20 to video decoder 30 in the encoded video bitstream, e.g., via a slice header, parameter set, or SEI message, and so a limit on the intended region based on LCU size may also be effectively signaled to video decoder 30. In some examples, video encoder 20 limits a size, e.g., height 138 or width 140 to the left of current video block 122, of intended region 128 based on an integer number of pixels. For example, video encoder 20 may limit width 140 to the left of current video block 122 to an integer number of pixels, such as 64, which may correspond to the width of LCUs 134 and 136.

[0104] Video encoder 20 may limit intended region 128 to include reconstructed video blocks without in-loop filtering, such as deblocking and sample adaptive offset (SAO) filtering. In this manner, video encoder 20 may limit two-dimensional vectors for IntraBC such that they may refer, in addition to current LCU 134, only to blocks in the

reconstructed area of a neighboring LCU, such as left LCU 136, for which in-loop filtering was not performed. Video encoder 20 could increase the size of intended region 128, but such increases may delay in-loop filtering and further processing of video blocks in the intended region, or require additional memory to store samples prior to in-loop filtering. Accordingly, video encoder 20 may limit intended region 128, e.g., as described herein, to achieve a balance between prediction accuracy and efficiency.

[0105] FIG. 4 is a conceptual diagram an example of a current block of video data 260 and neighboring blocks of video data 262, 264, 266 from which candidate predictive vectors for the current block of video data may be derived. In some examples, similar to merge mode or Advanced Motion Vector Prediction (AMVP) mode for inter prediction, video coders may determine the predictive two-dimensional vector for a current video block, e.g. PU, based on the two-dimensional vectors used to predict neighboring video blocks, e.g., PUs, using IntraBC. The neighboring blocks illustrated in FIG. 4 include a left neighboring block 262, above (top) neighboring block 264, and above-left (top-left) neighboring block 266. The neighboring blocks illustrated in FIG. 4 are one example. In other examples, a video coder may consider the vectors of more, fewer and/or different neighboring blocks, such as any neighboring block that may be considered according to the merge and AMVP modes for inter-prediction.

[0106] The predictor may be the horizontal and/or vertical displacement component of the two-dimensional block vector from a selected one of the neighboring blocks. In some examples, the predictor is always from a specific neighboring unit, e.g. the top one or the left one. In some examples, which of the neighboring blocks provides the predictive two-dimensional block vector for a current PU is determined based on the index of the PU and/or the PU shape. In some examples, the predictor can be a function (such as a mean or median) of the horizontal and/or vertical components of the two-dimensional block vector from a plurality of neighboring units.

[0107] In general, if a neighboring block cannot provide a predictive two-dimensional vector, then the predictive two-dimensional vector (or candidate predictive two-dimensional vector) may be set to zero or a default two-dimensional vector. In some examples, the vectors of neighboring blocks may only be available as predictive vectors for a current video block when the neighboring video block was predicted according to IntraBC. In other words, neighboring blocks not coded with the IntraMC mode may be considered unavailable for block vector prediction for the current video block. For example, when the predictor is always from the left neighboring block and the left

neighboring block was predicted with IntraMC mode, its two-dimensional vector is used as the predictive two-dimensional vector. When the predictor is always from the left neighboring block and the left neighboring block was not predicted with IntraMC mode, video coders may use a zero vector or default vector as the predictive two-dimensional vector for the current video block.

[0108] In some examples, a video coder may consider the two-dimensional vector of neighboring video blocks unavailable for prediction of the two-dimensional vector of the current video block if it is not in the in the same LCU or other larger or largest coding unit as the current video block, e.g., CU or PU. In some examples, if the current two-dimensional vector for the current video block e.g., PU or CU, is the first one of the current CU or LCU, the predictive two-dimensional vector for the current video block may be set to zero or a default two-dimensional vector. It is possible that the upper neighboring CU is considered to be unavailable if the upper neighboring CU and the current CU are not in the same LCU. It is also possible that the left neighboring CU is considered to be unavailable if the left neighboring CU and the current CU are not in the same LCU.

[0109] In some examples, in a manner similar to merge mode and AMVP mode for inter-prediction, video encoder 20 and video decoder 30 may construct a set of candidate predictive vectors for the current video block that includes block vectors of a number of neighboring blocks. In such examples, video encoder 20 may encode, and video decoder 30 may decode an index that signals which candidate is used to provide the predictive two-dimensional vector for the current video block. In some examples video coders may determine the horizontal displacement component and vertical displacement component, respectively, of the predictive two-dimensional vector based on different candidate vectors, e.g., from different neighboring blocks. In such examples, video coders may code a respective index into the candidate list horizontal and vertical displacement components.

[0110] As one example, for a current CU coded according to the IntraBC mode, to code its block vector, video coders check the left and top neighboring CUs. If neither of them is available, e.g., was coded according to IntraBC, a video coder may determine that the predictive two-dimensional vector of the current CU is a zero vector (for both horizontal and vertical components). If only one of the neighboring CUs is available, e.g., was coded with IntraBC, the video coders use the vector that was used for prediction of the available neighboring CU as the predictive two-dimensional vector of

current two-dimensional vector of the current CU. If both neighboring CUs are available , e.g., was coded with IntraBC, and their two-dimensional vectors are the same, then this two-dimensional vector is used as the predictive two-dimensional vector. If both neighboring CUs are available, e.g., were coded with IntraBC, and their block vectors are different, then the video coders may code a flag or other syntax element to indicate whether the block vector from the top neighboring CU or the block vector from the left neighboring CU is used as the predictive two-dimensional vector for the current CU

[0111] In another example, video coders may define two or more default block vectors, which can be from (as a non-limiting example) the set (–w, 0), (-2w, 0), (-8, 0), (0,0), (0, 8), (0, -h), (0, -2h), where w and h are the width and the height of the current video block e.g., CU, and the first component is horizontal displacement and the second component is vertical displacement. If both the left neighboring CU and the upper neighboring CU are available, the video coders may use their vectors as the first and the second predictive two-dimensional vectors for the current CU. If one of them is not available, video coders may use a default predictive vector to replace the unavailable predictive vector in a list of candidate predictive vectors. If both of them are not available, the video coders may use two default predictive vectors, which may be the same or different, to replace the unavailable predictive vector in a list of candidate predictive vectors. An advantage of replacing unavailable predictive vectors with default vectors in a candidate list is that a flag for the selection between predictors may also be included in the encoded video bitstream, so that video decoder 30 does not need to conditionally parse this flag. Although examples are described herein with respect to a candidate list including two predictive vectors derived from two neighboring blocks, if the blocks/vectors are available, in other examples a video coder may consider more or fewer neighboring blocks, and include more or fewer predictive two-dimensional vectors in a candidate list.

[0112] In some examples, for the first video block and corresponding two-dimensional vector used for IntraBC in each LCU, the video coders may use a different derivation process for determining its predictive two-dimensional vector than for other blocks and associated vectors within the LCU. For example, a video coder, e.g., video encoder 20 and/or video decoder 30, may determine that the predictive two-dimensional vector for the first video block and corresponding two-dimensional vector used for IntraBC in the LCU is a default two-dimensional vector. The default two-dimensional vector may be

(–w, 0), where w is the width of the current video block e.g., CU. In other examples, the default two-dimensional vector may be (-2w, 0), (-8, 0), (0,0), (0, 8), (0, -h), (0, -2h), where w and h are the width and the height of the current video block, e.g., CU. In other examples, a video coder, e.g., video encoder 20 and/or video decoder 30, may derive the predictive two-dimensional vector for the first video block and corresponding two-dimensional vector used for IntraBC in the LCU. In some examples, video coders may derive predictive two-dimensional vectors for the first video block coded using IntraBC in an LCU differently for video blocks, e.g., CUs, with different sizes or at different position in the LCU, or in other words may derive predictive two-dimensional vectors for the first video block coded using IntraBC in an LCU differently based on video block size or position in the LCU.

[0113] In some examples, video coders may select the method or methods used to determine the horizontal and vertical components of the two-dimensional vector of the current video block can be based on flags, syntax elements, or based on other information (such as the specific color space (e.g., YUV, RGB, or the like), the specific color format (e.g., 4:2:0, 4:4:4, or the like), the frame size, the frame rate, or the quantization parameter (QP), or based on previously coded frames).

[0114] In some examples, video encoder 20 may encode a flag, e.g., an IntraMC_flag or IntraBC_flag, to indicate to video decoder 30 whether a video block, e.g., CU, is encoded using the mode for predicting current video blocks based on predictive video blocks in the same picture, which may be referred to as the IntraMC or IntraBC mode. The flag may be a 1-bit flag whose value indicates whether or not one or more CUs, or other blocks or units, are encoded using the IntraMC or IntraBC mode, e.g., as described herein. Video coders may code a value of the flag for each video block, e.g., CU or PU, or may code a value of the flag to each of a plurality of video blocks, e.g., within a slice, or within a frame or picture, or sequence of pictures. Video coders may code the flag in, as examples, a slice header, a parameter set (e.g., PPS), or an SEI message.

[0115] Video coders may code the flag in bypass mode, or arithmetic encode, e.g., CABAC encode, the flag with context. In some examples, video coders may arithmetic code, e.g., CABAC encode, the flag with a single, fixed context that does not depend on neighboring video blocks. In other examples, video coders may arithmetic code, e.g., CABAC encode, the flag with a context derived from neighboring video blocks.

[0116] Referring to FIG. 4, when video coders code the flag for a current video block 260 using CABAC context, the value of context can be derived from the neighboring video blocks such as the top (above) neighboring video block 264, the top- left (above-left) neighboring video block 266, or the left neighboring video block 262. Video coders may consider neighboring video blocks, such as top neighboring video block 264, available for deriving a context for the current video block when it is within the same LCU as the current video block, and unavailable when the neighboring video block is out of the current LCU.

[0117] As an example, for a current video block, video coders may derive the context value = (top IntraMC_flag == 0) ? 0 : 1 + (left IntraMC_flag == 0) ? 0 : 1. In some examples, video coders may always derive the context for current video block 260 from a specific one of neighboring video blocks 262, 264, 266, e.g., left neighboring video block 262, and the context value for the current video block is (left IntraMC_flag == 0) ? 0 : 1. In some examples, video coders code the flag using a context value that depends on the size of the video block, e.g., CU. In some examples, video blocks with different sizes have different context values. In other words, each video block, e.g., CU, size may be associated with a respective, unique, context value. In some examples, several video blocks, e.g., CUs, with different predefined sizes share the same context value, and video blocks with other sizes use one or more different context values. In other words, video coders may associate each of a plurality of context values with a respective set or range(s) of possible video block, e.g., CU, sizes.

[0118] In some examples, video coders, e.g., video encoder 20 and/or video decoder 30, may code the horizontal displacement component and vertical displacement component of a two-dimensional vector for IntraBC (e.g., horizontal displacement component 112 and vertical displacement component 110 of two-dimensional vector 106 in FIG. 2) based on unary codes. In other examples, video coders may code the horizontal displacement component and vertical displacement component based on exponential Golomb or Rice-Golomb codes.

[0119] In some examples, the horizontal and vertical displacement components may only indicate regions above and to the left of the current video block, and the video coders may not need to retain or code sign bits for these values. In some examples, video coders may construct a frame of reference such that the areas above and to the left of the current video block may represent positive directions relative to the current video block. In such examples, if only the video blocks above and/or to the left of the current

video block are considered as candidate predictive video blocks, video coders may not need to retain or code sign bits because it may be pre-defined that all values of the horizontal displacement component and vertical displacement component represent positive (or negative) values and indicate video blocks above and/or to the left of the current video block.

[0120] In some examples, the maximum size of these two-dimensional vectors (or the difference between one or more two-dimensional vectors, such as a residual two-dimensional vector) may be small, e.g., due to the definition of the intended region and/or pipeline constraints, as discussed above. In such examples, video encoder 20 may binarize these two-dimensional block vectors with truncated values. In some examples, video encoder 20 may employ truncated unary, truncated exponential-golomb, or truncated golomb-rice codes in entropy encoding the two-dimensional vectors, e.g., in encoding the horizontal displacement component and vertical displacement component of the two-dimensional vectors.

[0121] The truncation value that video encoder 20 may use in any of the various truncated encoding schemes described herein can be constant, e.g., based on the LCU size. In some examples, the truncation value may be the same for the horizontal displacement component and vertical displacement component. In other examples, the truncation value may be different for horizontal displacement component and vertical displacement component.

[0122] As one illustrative example, if the size of an LCU is 64, e.g., 64x64, and the vertical components of the two-dimensional vectors are limited to be within the LCU, e.g., as described above with respect to FIG. 3, then the truncation can be equal to 63 for the horizontal component of the two-dimensional vector, and equal to 63 – MinCUSize for the vertical component of the two-dimensional vector. In some examples, the truncation value can be adaptive depending on the position of the current video block within the LCU. For example, if the vertical component of the two-dimensional vector is limited to be within the LCU, then video encoder can truncate the vector binarization to the difference between the top position of the current video block and the top position of the LCU.

[0123] Video encoder 20 may entropy encode the binarizations of the horizontal and vertical components of the two-dimensional vector using bypass mode, or may arithmetic encode the binarizations, e.g., with CABAC context. For example, video encoder 20 limits the search for a predictive video block to an intended region, e.g., as

illustrated in Fig. 3, the distribution of the components of the two dimensional vector (MV) may not be zero-centered. For example, MV_x tends to be negative since pixels on the right of the current video block, e.g., CU, (in the same row) have not been encoded/reconstructed. Similarly, MV_y tends to be negative since pixels below the current video block, e.g., CU, (in the same column) have not been encoded/reconstructed.

[0124] Bypass mode may assume equal probability for 0 and 1. For sign, this means that bypass mode assumes an equal probability of being positive or negative. Because the components of the two-dimensional vector do not have equal probability of being positive or negative, video encoder 20 may arithmetic encode, e.g., CABAC encode, the sign with context, e.g., with an initial probability other than 0.5.

[0125] One example of how video encoder 20 may encode the horizontal component of the two-dimensional vector (MV_x) is as follows. Although described with respect to MV_x, the technique could also be applied to encode the vertical component of the two-dimensional vector (MV_y), or the horizontal or vertical components of a residual two-dimensional vector that represents the difference between the current vector and a predictive vector, e.g., mvd_x and mvd_y.

[0126] MV_x may be represented by a sign value, and a binarization string (for abs(MV_x)) b0 b1 …. The first bin b0 indicates if abs(Mv_x) > 0 (b0=1) or not (b0 = 0). Video encoder 20 may encoder the first bin b0 using CABAC with a context. The b0 for Mv_x and Mv_y may have separate contexts, or may share the same contexts. In some examples, the i-th bin in MV coding of IntraBC share the same contexts with the i-th bins in block vector coding of Inter motion compensation. In some examples, the i-th bins in MV coding of IntraBC and block vector coding of Inter motion compensation do not share contexts.

[0127] The following bins b1b2… represent the value of abs(MV_x) -1, and video encoder 20 may encode these bins using Exponential Golomb codes with parameter 3 in bypass mode. In some examples, video encoder 20 uses other orders of Exponential Golomb codes, e.g., 1, 2, 4, 5. In some examples, b1 represents if abs(MV_x) = 1 (b1 = 1) or not (b1 = 0). In some examples, video encoder 20 may encode b1 with bypass mode or with CABAC context.

[0128] In some examples, b2b3… represent the value of abs(MV_x) -2, and video encoder 20 may encode these bins using Exponential Golomb codes with parameter 3, or other orders of Exponential Golomb codes, in bypass mode. The last bin may

indicate the sign of MV_x, and video encoder 20 may encode this bin in bypass mode without any context. In other examples, video encoder 20 may encode the sign bin encoded using CABAC with one or multiple contexts. The sign bins for MV_x and MV_y may have separate contexts, or it is possible that they share the same contexts.

[0129] In some examples, video coders, e.g., video encoder 20 and/or video decoder 30, may limit application of the IntraBC mode, and any associated coding of flags, vector components, or other syntax, to video blocks of a certain size, i.e., that meet a certain size criteria. The size criteria may be one or both of a maximum block size and/or a minimum block size (e.g., IntraMC_MaxSize and IntraMC_MinSize). Video encoder 20 may indicate the one or more size criteria to video decoder 30 in the encoded video bitstream, e.g., using flags or other syntax elements, or video coders may implicitly determine the size criteria based on other information, such as the specific color space (e.g., YUV, RGB, or the like), the specific color format (e.g., 4:2:0, 4:4:4, or the like), the frame size, the frame rate, the quantization parameter (QP), or based on previously coded frames. For example, IntraBC for small sizes might have an impact on the memory bandwidth requirements of the system, and video coders may apply a minimum block size criteria to restrict IntraBC to blocks above a size threshold.

[0130] FIGS. 5A and 5B show an example of a CU that is divided into two PUs (i.e. PU0 and PU1). FIGS. 5A and 5B show an illustration image of spatial neighbor derivation for HEVC AMVP and merge. The spatial neighbors may be used for block vector coding, such as for determining candidates for an Intra BC AMVP mode or an Intra BC merge mode. Video encoder 20 and/or video decoder 30 may derive spatial MV candidates from the neighboring blocks as shown in FIGS. 5A and 5B, for a specific PU (PU0), although the methods generating the candidates from the blocks may differ for merge and AMVP modes. In merge mode, video encoder 20 and/or video decoder 30 may derive up to four spatial MV candidates using the ordering shown in FIG. 5A. The order is as follows: left (0), above (1), above right (2), below left (3), and above left (4), as shown in FIG. 5A.

[0131] In AVMP mode, the neighboring blocks are divided into two groups: left group consisting of the block 0 and 1, and above group consisting of the blocks 2, 3, and 4 as shown in FIG. 5B. For each group, the potential candidate in a neighboring block referring to the same reference picture as that indicated by the signaled reference index has the highest priority to be chosen to form a final candidate of the group. It may be possible that all neighboring blocks do not contain a block vector pointing to the same

reference picture. Therefore, if such a candidate cannot be found, the first available candidate may be scaled to form the final candidate, thus the temporal distance differences can be compensated.

[0132] The current techniques for Intra BC may have some potential problems. In one example, when the chroma sampling format is not 4:4:4 (i.e., is either 4:2:0 or 4:2:2), a chroma motion compensation block size may be as small as 2x2, and thus smaller than the smallest block size as supported in the base HEVC specification. When accessing reference blocks, a video coder typically reads the blocks in the units of predefined memory blocks (e.g. 4x2, 8x2). Thus, if the block size is smaller than the predefined memory block units, extra pixels, which may not be used in the compensation, are accessed, thus lowering memory access efficiency.

[0133] As another example of a potential problem, simply scaling a luma block vector for a chroma component may result in a block vector pointing to a region within the current CU or PU. As another example, AMVP fashion of block vector prediction is not well supported in existing implementations of Intra BC.

[0134] This disclosure introduces techniques that may improve Intra BC prediction. According to one technique of this disclosure, when the coded video data has a chroma sampling format of 4:2:0, video decoder 30 may use a different block vector to predict a chroma component than the block vector used to predict a corresponding luma component. For example, a block vector for a first luma component and its corresponding first chroma component may be the same, but the block vector for a second chroma component may not be same as the block vector for its corresponding second luma component. Rather, the block vector for the second chroma component may be the same as the block vector for the first luma component.

[0135] Video decoder 30 may derive the block vector for chroma from the block vector used for luma. For example, for a 2Nx2N CU divided into four NxN PUs, video encoder 20 may signal four BVs, i.e. BV_1, $BV_{,2}$, BV_3, and BV_4 in coding order, for the four PUs/TUs in the current CU. According to techniques of this disclosure, video decoder 30 may use the four BVs for Intra BC prediction of the luma component of the current CU and for derivation of one single block vector for the IntraBC prediction of multiple chroma blocks of the whole CU. Thus, a CU containing four PUs/TUs may share one common block vector (denoted as BV_c) for the chroma blocks of multiple PUs.

[0136] Video decoder 30 may generate a common chroma block vector from the four luma block vectors, converting the luma block vector to a chroma block vector as done in HEVC version 1, or directly using the luma block vector as the chroma block vector, before motion compensation for the chroma components. Video decoder 30 may derive the luma block vector BV_L by one of the following methods:

1. Video decoder 30 selects the minimum block vector BV_L among BV_1, BV_2, BV_3, BV_4, and converts BV_L, for example by scaling, to BV_c. For purposes of explanation, a block vector BV_a is defined to be smaller than the other BV_b in one of the following possible ways.
    a. If the horizontal component of BV_a ($BV_a[0]$) is smaller than that of BV_b
    b. If the absolute value (denoted as |x|, for a value a) of the horizontal component of BV_a is smaller than that of BV_b
    c. If the $|BV_a[0]|+ |BV_a[1]|$ is smaller than that of $|BV_b[0]|+ |BV_b[1]|$
    d. If the vertical component of BV_a ($BV_a[1]$) is smaller than that of BV_b
2. Similar to the above minimum block vector selections, video decoder 30 may select a maximum block vector among the four block vectors, with one of various definitions of one vector being smaller than the other as defined above.
3. Video decoder 30 may determine an average block vector, by averaging the horizontal and vertical components for the four luma block vectors to derive the luma block vector BV_L, wherein a rounding of adding +2/-2 may or may not apply during the averaging. BV_L may be scaled to determine the common chroma block vector.
4. Video decoder 30 may select one block vector from BV_1, BV_2, BV_3, and BV_4 as BV_L. The chosen block vector may, for example, be a fixed one among the four, such as always BV_1, always BV_2, etc. or may be explicitly signaled in the bitstream.

[0137] According to another technique of this disclosure, when the coded video data has a chroma sampling format of 4:2:2, video decoder 30 may use a different BV for chroma than the BV used for luma. Instead, video decoder 30 may derive the BV used for chroma from the BV(s) used for luma. For example, IntraBC may be extended to PU/TU and suppose the PU/TU is NxN for a 2Nx2N CU, then there are two BVs, i.e. BV_1, and $BV_{,2}$, in coding order, for two PUs/TUs in the current CU. Similar to the

techniques described above, a minimum vector, maximum vector, or average vector may be generated from the two block vectors of the two PU/TUs of the current PU for the chroma block of the whole CU.

[0138] According to one technique of this disclosure, for a 2Nx2N CU that is coded with four NxN Intra BC PU/TUs, video decoder 30 may predict the chroma component with a single block vector generated from the block vectors of the NxN PU/TUs when the chroma sampling format is not 4:4:4. Video decoder 30 may, however, still use four separate block vectors for predicting the luma components of the PUs. For example, for a 2Nx2N CU that is divided into four NXN PUs, video encoder 20 may signal four separate block vectors, one for each PU of the CU. Video decoder 30 may decode those four separate block vectors and use the four separate block vectors to predict the luma components of the four PUs. For the chroma components of the four PUs, however, video decoder 30 may select one of the four block vectors and use that one block vector to predict the chroma components for more than one of the PUs of the CU. In some instances, video decoder 30 may use one block vector to predict the chroma components for all four PUs of the CU.

[0139] In another example, for a 2Nx2N CU that is coded with four NxN Intra BC PU/TUs, video decoder 30 may predict the chroma component with two block vectors generated from the block vectors of the NxN PU/TUs when the chroma sampling format is other than 4:4:4. Each of the two block vectors may be used to predict one N/2xN block or NxN/2 block. In other words, each of the two block vectors may be used to predict two of the four PUs of the CU. Video decoder 30 may check at least one of the four block vectors of the PUs of the CU is checked and the two luma block vectors are derived depending on these block vectors. More specifically, for example, the four block vectors are compared and the two neighboring block vectors with closer distance than any other pair may be chosen.

[0140] In one example, the dis(BV_0, BV_1), dis(BV_0, BV_2), dis(BV_1, BV_3) and dis(BV_2, BV_3) are calculated. (1) The smallest one corresponds to a pair that may be combined to a 2NxN or Nx2N block. For example, if dis(BV_0, BV_1) is the smallest, the top two NxN blocks are combined to a 2NxN block. The remaining ones automatically become the other 2NxN block. (2) The biggest one corresponds to a pair which may not be combined to a 2NxN or Nx2N block. For example, if dis(BV_0, BV_1) is the biggest, the left two NxN blocks are combined to a 2NxN block. The remaining ones automatically become the other 2NxN block. (3) For each combined 2NxN block, the two block

vectors are processed and to derive a single luma block vector. (4) The distance of the two block vectors may be simply defined as the absolute value of (BV_a[0]-BV_b [0]) plus the absolute value of (BV_a[1]-BV_b[1]) for two block vectors BV_a and BV_b.. Alternatively, other distance definitions that return non-negative values may also apply.

[0141] According to another technique of this disclosure, when video data is coded according to the 4:2:0 or 4:2:2 format, video decoder 30 may derive one motion from the four block vectors of the four NxN PU/TUs. In one example, video decoder 30 may derive a block vector for the bottom-right block (i.e. the bottom-right PU) and use this block vector to predict the four PUs of the CU. By selecting the block vector of the bottom-right PU, it may be ensured that the block vector does not point to another PU of the CU.

[0142] In other examples, video decoder 30 may select the block vector of the bottom-left block, top-right block, top-left block is chosen to be the only one to generate the block vector for the chroma blocks of this CU.

[0143] According to another technique of this disclosure, when video data is coded according to the 4:2:2 format, one block vector may be derived from two block vectors. In such an implementation, for a 2Nx2N CU that is divided into four NxN PUs, video decoder 30 may use the same block vector to predict two the four PUs of the CU. In one example, video decoder 30 may use the block vector of the top-right block (i.e. top-right PU) to predict both the top-right PU and the top-left PU. In this example, video decoder 30 may use the block vector of the bottom-right block (i.e. bottom-right PU) to predict the bottom-left PU and the bottom-right PU. In another example, video decoder 30 may use the block vector of the top-left block to predict the top two chroma blocks of the CU and use the block vector of the bottom-left block to predict the bottom two chroma blocks.

[0144] When video data is coded according to the 4:2:0 or 4:2:2 format and when a 2Nx2N CU is divided into 2NxN or Nx2N IntraBC PU/TUs, video decoder 30 may predict the chroma blocks of the PU/TUs with different block vectors. According to another technique of this disclosure, video decoder 30 may derive one block vector from the two luma block vectors and use the one block vector to predict both chroma blocks of the whole CU. In one example, video decoder 30 may select the block vector of the right block (when coded with Nx2N) or the block vector of the bottom block (when coded with 2NxN) as the block vector for the chroma blocks of the CU. In another example, video decoder 30 may select the block vector of the left block (when coded

with Nx2N) or the block vector of the top block (when coded with 2NxN) as the block vector for predicting the chroma blocks of the CU.

[0145] Regardless of how video decoder 30 derives a block vector from the luma PUs (denoted the derived block vector as dBV), to form the final chroma block vector, video decoder 30 may clip the derived luma block vector to produce a legal block vector. In one example, video decoder 30 may perform the clipping in a way that the chroma block vector is not pointing to a pixel within the same PU/TU. In another example, video decoder 30 may perform the clipping in a way that the chroma block vector is not pointing to a pixel within the same CU.

[0146] For a case where a 2Nx2N CU is divided into four NxN PUs, video decoder 30 may perform a horizontal clipping operation and a vertical clipping operation. Video decoder 30 may apply horizontal clipping as follows:

Horizontal clip: max(-leftW-(CUx%ctbW), dBV[0]), wherein

1. leftW could be a fix value of e.g., 64, or be the width of the current CTU, or any other value transmitted in the bitsream e.g., in the high level syntax, such as video parameter set, sequence parameter set and picture parameter set.
2. CUx is the horizontal coordinate of the top-left pixel of the current CU.
3. ctbW is the width of the CTB

Vertical clip: max (-CUy% ctbH, dBV[1]), wherein

1. CUy is the vertical coordinate of the top-left pixel of the current CU.
2. ctbH is the height of the CTB.

[0147] According to some examples, video decoder 30 may, for example, perform only horizontal clipping when a 2Nx2N CU is divided into Nx2N PUs or only perform vertical clipping when a 2Nx2N CU is divided into 2NxN PUs. In instances, as described above, when two block vectors from two NxN blocks are to be combined for an Nx2N block, video decoder 30 may apply only horizontal clipping. In instances, as described above, when two block vectors from two NxN blocks are to be combined for a 2NxN block, video decoder 30 may apply only vertical clipping.

[0148] In some implementations, the various processes described above may only be implemented for certain CU sizes. For example, video decoder 30 may only use a single block vector to predict multiple chroma blocks for CU sizes equal to 8x8 or smaller. The techniques described herein may, for example, only apply for CU sizes

that are not larger than a predefined size, such as 32x32, 16x16, or 8x8. This size may be predefined, or alternatively, may be signaled at sps/pps/slice level.

[0149] FIG. 6 is a block diagram illustrating an example video encoder that may implement the techniques described in this disclosure. Video encoder 20 may perform intra- and inter-coding of video blocks within video slices. Intra coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame or picture. Inter-coding relies on temporal or inter-view prediction to reduce or remove redundancy in video within adjacent frames or pictures of a video sequence. Intra-mode (I mode) may refer to any of several spatial based compression modes. Inter-modes, such as uni-directional prediction (P mode) or bi-prediction (B mode), may include to any of several temporal-based compression modes. Video encoder 20 may also be configured to utilize a mode for intra prediction of blocks of video data from predictive blocks of video data within the same picture, e.g., an IntraBC or IntraMC mode, as described herein.

[0150] In the example of FIG. 6, video encoder 20 includes video data memory 334, a partitioning unit 335, prediction processing unit 341, intended region memory 364, filter processing unit 366, reference picture memory 368, summer 350, transform processing unit 352, quantization processing unit 354, and entropy encoding unit 356. Prediction processing unit 341 includes motion estimation unit 342, motion compensation unit 344, intra prediction processing unit 346, and Intra Block Copy (IntraBC) unit 348, and. For video block reconstruction, video encoder 20 also includes inverse quantization processing unit 358, inverse transform processing unit 360, and summer 362.

[0151] In various examples, a unit of video encoder 20 may be tasked to perform the techniques of this disclosure. Also, in some examples, the techniques of this disclosure may be divided among one or more of the units of video encoder 20. For example, IntraBC unit 348 may perform the techniques of this disclosure, alone, or in combination with other units of video encoder, such as motion estimation unit 342, motion compensation unit 344, intra prediction processing unit 346, intended region memory 364, and entropy encoding unit 356.

[0152] Video data memory 334 may store video data to be encoded by the components of video encoder 20. The video data stored in video data memory 334 may be obtained, for example, from video source 18. Intended region memory 364 may be a reference block memory that stores reference video data for use in encoding video data by video encoder 20, e.g., in intra- or inter-coding modes. Video data memory 334 and intended

region memory 364 may be formed by any of a variety of memory devices, such as dynamic random access memory (DRAM), including synchronous DRAM (SDRAM), magnetoresistive RAM (MRAM), resistive RAM (RRAM), or other types of memory devices. Video data memory 334 and intended region memory 364 may be provided by the same memory device or separate memory devices. In various examples, video data memory 334 may be on-chip with other components of video encoder 20, or off-chip relative to those components.

[0153] Partitioning unit 335 partitions the data from video data memory 334 into video blocks. This partitioning may also include partitioning into slices, tiles, or other larger units, as well as video block partitioning, e.g., according to a quadtree structure of LCUs and CUs. Video encoder 20 generally illustrates the components that encode video blocks within a video slice to be encoded. The slice may be divided into multiple video blocks (and possibly into sets of video blocks referred to as tiles).

[0154] Prediction processing unit 341 may select one of a plurality of possible coding modes, such as one of a plurality of intra coding modes, one of a plurality of inter coding modes, or an IntraBC mode according to the techniques described in this disclosure, for the current video block based on error results (e.g., coding rate and the level of distortion). Prediction processing unit 341 may provide the resulting predictive block to summer 250 to generate residual block data and to summer 262 to reconstruct the current block for use in prediction of other video blocks, e.g., as a reference picture.

[0155] Intra prediction processing unit 346 within prediction processing unit 341 may perform intra-predictive coding of the current video block relative to one or more neighboring blocks in the same frame or slice as the current block to be coded to provide spatial compression. Motion estimation unit 342 and motion compensation unit 344 within prediction processing unit 341 perform inter-predictive coding of the current video block relative to one or more predictive blocks in one or more reference pictures, e.g., to provide temporal compression.

[0156] Motion estimation unit 342 may be configured to determine the inter-prediction mode for a video slice according to a predetermined pattern for a video sequence. Motion estimation unit 342 and motion compensation unit 344 may be highly integrated, but are illustrated separately for conceptual purposes. Motion estimation, performed by motion estimation unit 342, is the process of generating motion vectors, which estimate motion for video blocks. A motion vector, for example, may indicate the displacement of a PU of a video block within a current video frame or picture relative to

a predictive block within a reference picture. Similarly, the two-dimensional vectors used for IntraBC according to the techniques of this disclosure indicate the displacement of a PU of a video block within a current video frame or picture relative to a predictive block within the same frame or picture. IntraBC unit 348 may determine two-dimensional vectors, e.g., block vectors or motion vectors, for IntraBC coding in a manner similar to the determination of motion vectors by motion estimation unit 342 for inter prediction, or may utilize motion estimation unit 342 to determine the two-dimensional vectors.

[0157] A predictive block, e.g., identified by motion estimation unit 342 and/or IntraBC unit for inter prediction or IntraBC prediction, is a block that is found to closely match the PU of the video block to be coded in terms of pixel difference, which may be determined by sum of absolute difference (SAD), sum of square difference (SSD), or other difference metrics. In some examples, video encoder 20 may calculate values for sub-integer pixel positions of reference pictures stored in reference picture memory 368 or the intended region of the current picture stored in intended region memory 364. For example, video encoder 20 may interpolate values of one-quarter pixel positions, one-eighth pixel positions, or other fractional pixel positions of the reference picture. Therefore, motion estimation unit 342 and/or IntraBC unit 348 may perform a search for a predictive block relative to the full pixel positions and fractional pixel positions and output a vector with fractional pixel precision.

[0158] Motion estimation unit 342 calculates a motion vector for a PU of a video block in an inter-coded slice by comparing the position of the PU to the position of a predictive block of a reference picture. The reference picture may be selected from a first reference picture list (List 0 or RefPicList0) or a second reference picture list (List 1 or RefPicList1), each of which identify one or more reference pictures stored in reference picture memory 368. Motion estimation unit 342 sends the calculated motion vector to entropy encoding unit 356 and motion compensation unit 344.

[0159] Motion compensation, performed by motion compensation unit 344, may involve fetching or generating the predictive block based on the motion vector determined by motion estimation, possibly performing interpolations to sub-pixel precision. Upon receiving the motion vector for the PU of the current video block, motion compensation unit 344 may locate the predictive block to which the motion vector points in one of the reference picture lists.

[0160] In some examples, IntraBC unit 348 may generate two-dimensional vectors and fetch predictive blocks in a manner similar to that described above with respect to motion estimation unit 342 and motion compensation unit 344, but with the predictive blocks being in the same picture or frame as the current block. In other examples, IntraBC unit 348 may use motion estimation unit 342 and motion compensation unit 344, in whole or in part, to perform such functions for IntraBC prediction according to the techniques described herein. In either case, for IntraBC, a predictive block may be a block that is found to closely match the block to be coded, in terms of pixel difference, which may be determined by sum of absolute difference (SAD), sum of square difference (SSD), or other difference metrics, and identification of the block may include calculation of values for sub-integer pixel positions.

[0161] Whether the predictive video block is from the same picture according to IntraBC prediction, or a different picture according to inter prediction, video encoder 20 may form a residual video block by subtracting pixel values of the predictive block from the pixel values of the current video block being coded, forming pixel difference values. The pixel difference values form residual data for the block, and may include both luma and chroma difference components. Summer 250 represents the component or components that perform this subtraction operation. IntraBC unit 348 and/or motion compensation unit 344 may also generate syntax elements associated with the video blocks and the video slice for use by video decoder 30 in decoding the video blocks of the video slice. The syntax elements may include, for example, syntax elements defining the vector used to identify the predictive block, any flags indicating the prediction mode, or any other syntax described with respect to the techniques of this disclosure.

[0162] Intra prediction processing unit 346 may intra-predict a current block, as an alternative to the inter-prediction performed by motion estimation unit 342 and motion compensation unit 344, or the IntraBC prediction performed by IntraBC unit 348, as described above. In particular, intra prediction processing unit 346 may determine an intra-prediction mode to use to encode a current block. In some examples, intra prediction processing unit 346 may encode a current video block using various intra-prediction modes, e.g., during separate encoding passes, and intra prediction processing unit 346 (or prediction processing unit 341, in some examples) may select an appropriate intra-prediction mode to use from the tested modes. For example, intra prediction processing unit 346 may calculate rate-distortion values using a rate-

distortion analysis for the various tested intra-prediction modes, and select the intra-prediction mode having the best rate-distortion characteristics among the tested modes. Rate-distortion analysis generally determines an amount of distortion (or error) between an encoded block and an original, unencoded block that was encoded to produce the encoded block, as well as a bit rate (that is, a number of bits) used to produce the encoded block. Intra prediction processing unit 346 may calculate ratios from the distortions and rates for the various encoded blocks to determine which intra-prediction mode exhibits the best rate-distortion value for the block.

[0163] In any case, after selecting an intra-prediction mode for a block, intra prediction processing unit 346 may provide information indicative of the selected intra-prediction mode for the block to entropy encoding unit 356. Entropy encoding unit 356 may encode the information indicating the selected intra-prediction mode in accordance with the techniques of this disclosure. Video encoder 20 may include in the transmitted bitstream configuration data, which may include a plurality of intra-prediction mode index tables and a plurality of modified intra-prediction mode index tables (also referred to as codeword mapping tables), definitions of encoding contexts for various blocks, and indications of a most probable intra-prediction mode, an intra-prediction mode index table, and a modified intra-prediction mode index table to use for each of the contexts.

[0164] After prediction processing unit 341 generates the predictive block for the current video block via inter-prediction, intra-prediction, or IntraBC prediction, video encoder 20 forms a residual video block by subtracting the predictive block from the current video block, e.g., via summer 350. The residual video data in the residual block may be included in one or more TUs and applied to transform processing unit 352. Transform processing unit 352 transforms the residual video data into residual transform coefficients using a transform, such as a discrete cosine transform (DCT) or a conceptually similar transform. Transform processing unit 352 may convert the residual video data from a pixel domain to a transform domain, such as a frequency domain.

[0165] Transform processing unit 352 may send the resulting transform coefficients to quantization processing unit 354. Quantization processing unit 354 quantizes the transform coefficients to further reduce bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. The degree of quantization may be modified by adjusting a quantization parameter. In some examples, quantization

processing unit 354 may then perform a scan of the matrix including the quantized transform coefficients. Alternatively, entropy encoding unit 356 may perform the scan.

[0166] Following quantization, entropy encoding unit 356 entropy encodes the quantized transform coefficients. For example, entropy encoding unit 356 may perform context adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), syntax-based context-adaptive binary arithmetic coding (SBAC), probability interval partitioning entropy (PIPE) coding or another entropy encoding methodology or technique. Entropy encoding unit 356 may perform any of the techniques described herein for binarization and encoding syntax elements, including vector components, flags, and other syntax elements, for the prediction according to the IntraBC mode. Following the entropy encoding by entropy encoding unit 356, the encoded video bitstream may be transmitted to video decoder 30, or archived for later transmission or retrieval by video decoder 30.

[0167] Inverse quantization processing unit 358 and inverse transform processing unit 360 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual block in the pixel domain for later use as a reference block for prediction of other video blocks. Motion compensation unit 344 and/or IntraBC unit 348 may calculate a reference block by adding the residual block to a predictive block of one of the reference pictures within one of the reference picture lists. Motion compensation unit 344 and/or IntraBC unit 348 may also apply one or more interpolation filters to the reconstructed residual block to calculate sub-integer pixel values for use in motion estimation.

[0168] Summer 362 adds the reconstructed residual block to the motion compensated prediction block produced by motion compensation unit 344 and/or IntraBC unit 348 to produce a reconstructed video block. Intended region memory 364 stores reconstructed video blocks according to the definition of the intended region for IntraBC of a current video block by video encoder 20, e.g., IntraBC unit 348, as described herein. Intended region memory 364 may store reconstructed video blocks that have not been in-loop filtered by filter processing unit 366. Summer 362 may provide the reconstructed video blocks to filter processing unit 366 in parallel with intended region memory 364, or intended region memory 364 may release the reconstructed video blocks to filter processing unit 366 when no longer need for the intended region for IntraBC. In either case, IntraBC unit 348 may search the reconstructed video blocks in intended region

memory 364 for a predictive video block within the same picture as the current video block to predict the current video block.

[0169] Filter processing unit 366 may perform in-loop filtering on the reconstructed video blocks. In-loop filtering may include deblock filtering to filter block boundaries to remove blockiness artifacts from reconstructed video. In-loop filtering may also include SAO filtering to improve the reconstructed video. Reconstructed blocks, some of which may be in-loop filtered, may be stored in reference picture memory 368 as reference pictures. The reference pictures may include reconstructed blocks that may be used by motion estimation unit 342 and motion compensation unit 344 as a predictive blocks to inter-predict a block in a subsequent video frame or picture.

[0170] In this manner, video encoder 20 may be configured to implement the example techniques of this disclosure for intra prediction of blocks of video data from prediction blocks of video data within the same picture, e.g., according to an IntraBC mode. For example, video encoder 20 may be an example of a video encoder configured to perform a method of encoding video data including a mode for intra prediction of blocks of video data from predictive blocks of video data within the same picture, the method comprising selecting a predictive block of video data for a current block of video data from a set of previously encoded blocks of video data within the same picture as the current block of video data. The method further comprises determining a two-dimensional vector, wherein the two-dimensional vector has a horizontal displacement component and a vertical displacement component, wherein the horizontal displacement component represents a horizontal displacement between the predictive block of video data and the current block of video data and the vertical displacement component represents a vertical displacement between the predictive block of video data and the current block of video data. The method further comprises determining a residual block based on the current block of video data and the predictive block of video data, and encoding, in an encoded video bitstream, one or more syntax elements that define the horizontal displacement component and the vertical displacement component of the two-dimensional vector and the residual block

[0171] Video encoder 20 may also be an example of a video encoder that comprises a memory configured to store an encoded video bitstream, and one or more processors. The one or more processor are configured to select a predictive block of video data for a current block of video data from a set of previously encoded blocks of video data within the same picture as the current block of video data. The one or more processors are

further configured to determine a two-dimensional vector, wherein the two-dimensional vector has a horizontal displacement component and a vertical displacement component, wherein the horizontal displacement component represents a horizontal displacement between the predictive block of video data and the current block of video data and the vertical displacement component represents a vertical displacement between the predictive block of video data and the current block of video data. The one or more processor are further configured to determine a residual block based on the current block of video data and the predictive block of video data, and encode, in the encoded video bitstream, one or more syntax elements that define the horizontal displacement component and the vertical displacement component of the two-dimensional vector and the residual block.

[0172] FIG. 7 is a block diagram illustrating an example video decoder that may implement the techniques described in this disclosure. In the example of FIG. 7, video decoder 30 includes video data memory 378, an entropy decoding unit 380, prediction processing unit 381, inverse quantization processing unit 386, inverse transform processing unit 388, summer 390, an intended region memory 392, a filter processing unit 394, and reference picture memory 396. Prediction processing unit 381 includes motion compensation unit 382, intra prediction processing unit 384, and an Intra Block Copy (IntraBC) unit 385. Video decoder 30 may, in some examples, perform a decoding pass generally reciprocal to the encoding pass described with respect to video encoder 20 from FIG. 6.

[0173] In various examples, a unit of video decoder 30 may be tasked to perform the techniques of this disclosure. Also, in some examples, the techniques of this disclosure may be divided among one or more of the units of video decoder 30. For example, IntraBC unit 385 may perform the techniques of this disclosure, alone, or in combination with other units of video decoder 30, such as motion compensation unit 382, intra prediction processing unit 384, intended region memory 392, and entropy decoding unit 380.

[0174] During the decoding process, video decoder 30 receives an encoded video bitstream that represents video blocks of an encoded video slice and associated syntax elements from video encoder. Video data memory 378 may store video data, such as an encoded video bitstream, to be decoded by the components of video decoder 30. The video data stored in video data memory 378 may be obtained, for example, from computer-readable medium 16, e.g., from a local video source, such as a camera, via

wired or wireless network communication of video data, or by accessing physical data storage media. Video data memory 378 may form a coded picture buffer (CPB) that stores encoded video data from an encoded video bitstream. Intended region memory 392 may be a reference block memory that stores reference video data for use in decoding video data by video decoder 30, e.g., in intra-block coding modes. Video data memory 378 and intended region memory 392 may be formed by any of a variety of memory devices, such as DRAM, including SDRAM, MRAM, RRAM, or other types of memory devices. Video data memory 378 and intended region memory 392 may be provided by the same memory device or separate memory devices. In various examples, video data memory 378 may be on-chip with other components of video decoder 30, or off-chip relative to those components.

[0175] Entropy decoding unit 380 of video decoder 30 entropy decodes the bitstream to generate quantized coefficients, motion vectors for inter prediction, two-dimensional vectors for IntraBC prediction, and other syntax elements described herein. Entropy decoding unit 380 may perform the inverse of any of the techniques described herein for binarization and encoding syntax elements, including vector components, flags, and other syntax elements, for the prediction according to the IntraBC mode. Entropy decoding unit 380 forwards the vectors and other syntax elements to prediction processing unit 381. Video decoder 30 may receive the syntax elements at the sequence level, the picture level, the video slice level and/or the video block level.

[0176] When the video slice is coded as an intra-coded (I) slice, intra prediction processing unit 384 of prediction processing unit 381 may generate prediction data for a video block of the current video slice based on a signaled intra prediction mode and data from previously decoded blocks of the current frame or picture. When the video frame is coded as an inter-coded (i.e., B or P) slice, motion compensation unit 382 of prediction processing unit 381 produces predictive blocks for a video block of the current video slice based on the motion vectors and other syntax elements received from entropy decoding unit 280. The predictive blocks may be produced from one of the reference pictures within one of the reference picture lists. Video decoder 30 may construct the reference frame lists, RefPicList0 and RefPicList1, using default construction techniques or any other technique based on reference pictures stored in reference picture memory 396. When the video block is coded according to the IntraBC mode described herein, IntraBC unit 385 of prediction processing unit 381 produces predictive blocks for the current video block based on the two-dimensional vectors and

other syntax elements received from entropy decoding unit 380. The predictive blocks may be within an intended region within the same picture as the current video block defined by video encoder 20, and retrieved from intended region memory 392.

[0177] Motion compensation unit 382 and IntraBC unit 385 determine prediction information for a video block of the current video slice by parsing the vectors and other syntax elements, and uses the prediction information to produce the predictive blocks for the current video block being decoded. For example, motion compensation unit 282 uses some of the received syntax elements to determine a prediction mode (e.g., intra- or inter-prediction) used to code the video blocks of the video slice, an inter-prediction slice type (e.g., B slice or P slice), construction information for one or more of the reference picture lists for the slice, motion vectors for each inter-encoded video block of the slice, inter-prediction status for each inter-coded video block of the slice, and other information to decode the video blocks in the current video slice. Similarly, IntraBC unit 385 may use some of the received syntax elements, e.g., a flag, to determine that the current video block was predicted using the IntraBC mode, construction information for intended region memory 392 indicating which video blocks of the picture are within the intended region and should be stored in intended region memory, two-dimensional vectors for each IntraBC predicted video block of the slice, IntraBC prediction status for each IntraBC predicted video block of the slice, and other information to decode the video blocks in the current video slice.

[0178] Motion compensation unit 382 and IntraBC unit 385 may also perform interpolation based on interpolation filters. Motion compensation unit 382 and IntraBC unit 385 may use interpolation filters as used by video encoder 20 during encoding of the video blocks to calculate interpolated values for sub-integer pixels of predictive blocks. In this case, motion compensation unit 382 and IntraBC unit 385 may determine the interpolation filters used by video encoder 20 from the received syntax elements and use the interpolation filters to produce predictive blocks.

[0179] Inverse quantization processing unit 386 inverse quantizes, i.e., dequantizes, the quantized transform coefficients provided in the bitstream and decoded by entropy decoding unit 380. The inverse quantization process may include use of a quantization parameter calculated by video encoder 20 for each video block in the video slice to determine a degree of quantization and, likewise, a degree of inverse quantization that should be applied. Inverse transform processing unit 388 applies an inverse transform, e.g., an inverse DCT, an inverse integer transform, or a conceptually similar inverse

transform process, to the transform coefficients in order to produce residual blocks in the pixel domain.

[0180] After motion compensation unit 382 or IntraBC unit 385 generates the predictive block for the current video block based on the vectors and other syntax elements, video decoder 30 forms a decoded video block by summing the residual blocks from inverse transform processing unit 388 with the corresponding predictive blocks generated by motion compensation unit 382 and IntraBC unit 385. Summer 390 represents the component or components that perform this summation operation to produce reconstructed video blocks.

[0181] Intended region memory 392 stores reconstructed video blocks according to the definition of the intended region for IntraBC of a current video block by video encoder 20, as described herein. Intended region memory 392 may store reconstructed video blocks that have not been in-loop filtered by filter processing unit 394. Summer 390 may provide the reconstructed video blocks to filter processing unit 394 in parallel with intended region memory 392, or intended region memory 392 may release the reconstructed video blocks to filter processing unit 394 when no longer need for the intended region for IntraBC. In either case, IntraBC unit 385 retrieve a predictive video block for a current video block from intended region memory 392.

[0182] Filter processing unit 394 may perform in-loop filtering on the reconstructed video blocks. In-loop filtering may include deblock filtering to filter block boundaries to remove blockiness artifacts from reconstructed video. In-loop filtering may also include SAO filtering to improve the reconstructed video. Reconstructed blocks, some of which may be in-loop filtered, may be stored in reference picture memory 368 as reference pictures. The reference pictures may include reconstructed blocks that may be used by motion compensation unit 382 as a predictive blocks to inter-predict a block in a subsequent video frame or picture. Reference picture memory 396 also stores decoded video for later presentation on a display device, such as display device 32 of FIG. 1.

[0183] In this manner, video decoder 30 may be configured to implement the example techniques of this disclosure for intra predicting current blocks of video data based on a predictive blocks of video data within the same picture . For example, video decoder 30 may be an example of a video decoder configured to perform a method of decoding video data including a mode for intra prediction of blocks of video data from predictive blocks of video data within the same picture, the method comprising receiving, in an

encoded video bitstream, one or more syntax elements that define a horizontal displacement component and a vertical displacement component of a two-dimensional vector and a residual block for a current block of video data, and decoding the one or more syntax elements. The method further comprises determining a predictive block of video data for the current block of video data based on the decoded syntax elements that define the horizontal displacement component and the vertical displacement component of the two-dimensional vector, wherein the predictive block of video data is a reconstructed block of video data within the same picture as the current block of video data, and reconstructing the current block of video data based on the predictive block of video data and the residual block

[0184] Video decoder 30 may also be an example of a video decoder that comprises a memory configured to store an encoded video bitstream that encodes the video data, and one or more processors. The one or more processors of the video decoder may be configured to decode the one or more syntax elements, determine a predictive block of video data for the current block of video data based on the decoded syntax elements that define the horizontal displacement component and the vertical displacement component of the two-dimensional vector, wherein the predictive block of video data is a reconstructed block of video data within the same picture as the current block of video data, and reconstruct the current block of video data based on the predictive block of video data and the residual block.

[0185] FIG. 8 is a flowchart illustrating an example method for encoding video data, according to techniques described in this disclosure. The techniques of FIG. 8 will be described with reference to a video encoder, such as video encoder 20. Video encoder 20 determines from a plurality of vectors associated with the coding unit a vector for a first chroma block of a CU of a picture (402). Video encoder 20 locates a first chroma reference block using the vector (404). The first chroma reference block may be in the same picture as the CU. Video encoder 20 predicts the first chroma block based on the first chroma reference block (406). Video encoder 20 locates a second chroma reference block using the vector (408). The second chroma reference block may be in the same picture as the CU. Video encoder 20 predicts a second chroma block of the coding unit based on the second chroma reference block (410).

[0186] Video encoder 20 generates, for inclusion in the encoded video data, an indication that the CU of the picture is coded using an intra block copy mode and an indication of a vector for each luma block of the CU (412). As explained above, video

encoder 20 may predict each of the luma components using a separate block vector but predict the chroma components of the CU using a common block vector determined from the separate block vectors used to predict the luma components. Accordingly, video encoder 20 may include in the encoded video data an indication a separate block vector for each luma component of the CU.

[0187] FIG. 9 is a flowchart illustrating an example method for decoding video data, according to techniques described in this disclosure. The techniques of FIG. 9 will be described with reference to a video decoder, such as video decoder 30. Video decoder 30 determines a CU of a picture of the video data is coded using an intra block copy mode (420). Video decoder 30 determines from a plurality of vectors associated with the coding unit a vector for a first chroma block of the CU (422). Video decoder 30 locates a first chroma reference block using the vector (424). The first chroma reference block is in the same picture as the CU. Video decoder 30 predicts the first chroma block based on the first chroma reference block (426). Video decoder 30 locates a second chroma reference block using the vector (428). The second chroma reference block is in the same picture as the CU. Video decoder 30 predicts a second chroma block of the CU based on the second chroma reference block (430).

[0188] The CU may, for example, include two or more PUs, with the first chroma block corresponding to a first PU and the second chroma block corresponding to a second PU. Video decoder 30 may determine the vector for the second chroma block of the CU by obtaining a vector for the first PU.

[0189] In one example, the CU may include two or more PUs, with each PU having an associated vector. Video decoder 30 may locate a first luma reference block using the vector associated with the first prediction unit. The first luma reference block may corresponds to the first PU. Video decoder 30 may locate a second luma reference block using the vector associated with the second PU. The second reference luma block may correspond to the second PU. Video decoder 30 may locate the first chroma reference block using the vector associated with the first PU and locating the second chroma reference block using the vector associated with the first PU.

[0190] In one example, when a chroma sampling format for the CU is 4:2:0, the first chroma block may be a bottom-right block of the CU. The second chroma block may be a top-right block of the CU, and video decoder 30 may locate a third reference block using the vector. The third reference block may be in the picture. Video decoder 30 may predict a third chroma block based on the first reference block. The third chroma

block may be a top-left block of the CU. Video decoder 30 may locate a fourth reference block using the vector and predict a fourth chroma block of the CU based on the second reference block. The fourth reference block may be in the picture, and the fourth chroma block may be a bottom-left block of the CU.

[0191] In one example, a chroma sampling format for the CU may be 4:2:2. The first chroma block may be a top-right block, and the second chroma block may be a top-left block of the CU. Video decoder 30 may determine a second vector for a third chroma block of the CU and locate a third reference block using the second vector. The third reference block may be in the picture. Video decoder 30 may use the third reference block as a predictor for the third chroma block. Video decoder 30 may locate a fourth reference block using the vector. The fourth reference block may be in the picture. Video decoder 30 may use the fourth reference block as a predictor for a fourth chroma block of the CU.

[0192] It is to be recognized that depending on the example, certain acts or events of any of the techniques described herein can be performed in a different sequence, may be added, merged, or left out altogether (e.g., not all described acts or events are necessary for the practice of the techniques). Moreover, in certain examples, acts or events may be performed concurrently, e.g., through multi-threaded processing, interrupt processing, or multiple processors, rather than sequentially.

[0193] In one or more examples, the functions described may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the functions may be stored on or transmitted over as one or more instructions or code on a computer-readable medium and executed by a hardware-based processing unit. Computer-readable media may include computer-readable storage media, which corresponds to a tangible medium such as data storage media, or communication media including any medium that facilitates transfer of a computer program from one place to another, e.g., according to a communication protocol. In this manner, computer-readable media generally may correspond to (1) tangible computer-readable storage media which is non-transitory or (2) a communication medium such as a signal or carrier wave. Data storage media may be any available media that can be accessed by one or more computers or one or more processors to retrieve instructions, code and/or data structures for implementation of the techniques described in this disclosure. A computer program product may include a computer-readable medium.

[0194] By way of example, and not limitation, such computer-readable storage media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage, or other magnetic storage devices, flash memory, or any other medium that can be used to store desired program code in the form of instructions or data structures and that can be accessed by a computer. Also, any connection is properly termed a computer-readable medium. For example, if instructions are transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, DSL, or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. It should be understood, however, that computer-readable storage media and data storage media do not include connections, carrier waves, signals, or other transitory media, but are instead directed to non-transitory, tangible storage media. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk and Blu-ray disc, where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above should also be included within the scope of computer-readable media.

[0195] Instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated hardware and/or software modules configured for encoding and decoding, or incorporated in a combined codec. Also, the techniques could be fully implemented in one or more circuits or logic elements.

[0196] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, an integrated circuit (IC) or a set of ICs (e.g., a chip set). Various components, modules, or units are described in this disclosure to emphasize functional aspects of devices configured to perform the disclosed techniques, but do not necessarily require realization by different hardware units. Rather, as described above, various units may be combined in a codec hardware

unit or provided by a collection of interoperative hardware units, including one or more processors as described above, in conjunction with suitable software and/or firmware.

[0197] Various examples have been described. These and other examples are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A method of decoding video data, the method comprising:

determining a coding unit of a picture of the video data is coded using an intra block copy mode;

determining from a plurality of vectors associated with the coding unit a vector for a first chroma block of the coding unit;

locating a first chroma reference block using the vector, wherein the first chroma reference block is in the picture;

predicting the first chroma block based on the first chroma reference block;

using the same vector that was used to locate the first chroma reference block, locating a second chroma reference block for a second chroma block of the same coding unit, wherein the second chroma reference block is in the picture, and different from the first chroma reference block; and

predicting the second chroma block of the coding unit based on the second chroma reference block.

2. The method of claim 1, wherein the coding unit comprises two or more prediction units, wherein the first chroma block corresponds to a first prediction unit of the coding unit and the second chroma block corresponds to a second prediction unit of the coding unit, and wherein determining the vector for the first chroma block of the coding unit comprises obtaining a vector for the first prediction unit.

3. The method of claim 1, wherein the coding unit comprises two or more prediction units, wherein each prediction unit has an associated vector, the method further comprising:

locating a first luma reference block using the vector associated with a first prediction unit of the coding unit;

locating a second luma reference block using the vector associated with the second prediction unit of the coding unit;

locating the first chroma reference block using the vector associated with the first prediction unit; and;

locating the second chroma reference block using the vector associated with the first prediction unit.

4. The method of claim 1, wherein a chroma sampling format for the coding unit is 4:2:0, and wherein the vector determined from the plurality of vectors is associated with a bottom-right block of the coding unit.

5. The method of claim 4, wherein the second chroma block is a top-right block of the coding unit, the method further comprising:

locating a third reference block using the vector, wherein the third reference block is in the picture;

predicting a third chroma block based on the first reference block, wherein the third chroma block is a top-left block of the coding unit;

locating a fourth reference block using the vector, wherein the fourth reference block is in the picture;

predicting a fourth chroma block of the coding unit based on the second reference block, wherein the fourth chroma block is a bottom-left block of the coding unit.

6. The method of claim 1, wherein a chroma sampling format for the coding unit is 4:2:2, and wherein the first chroma block is a top-right block and the second chroma block is a top-left block of the coding unit.

7. The method of claim 6, further comprising:

determining a second vector for a third chroma block of the coding unit;

locating a third reference block using the second vector, wherein the third reference block is in the picture;

using the third reference block as a predictor for the third chroma block;

locating a fourth reference block using the vector, wherein the fourth reference block is in the picture;

using the fourth reference block as a predictor for a fourth chroma block of the coding unit.

8. The method of claim 1, wherein the coding unit comprises residual data.

9. A method of encoding video data, the method comprising:

determining from a plurality of vectors associated with the coding unit a vector for a first chroma block of a coding unit of a picture;

locating a first chroma reference block using the vector, wherein the first chroma reference block is in the picture;

predicting the first chroma block based on the first chroma reference block;

using the same vector that was used to locate the first chroma reference block, locating a second chroma reference block for a second chroma block of the same coding unit, wherein the second chroma reference block is in the picture, and different from the first chroma reference block;

predicting a second chroma block of the coding unit based on the second chroma reference block; and

generating, for inclusion in the encoded video data, an indication that the coding unit of the picture is coded using an intra block copy mode and an indication of a vector for each luma block of the coding unit.

10. The method of claim 9, wherein the coding unit comprises two or more prediction units and wherein the first chroma block corresponds to a first prediction unit and the second chroma block corresponds to a second prediction unit, and wherein determining the vector for the second chroma block of the coding unit comprises obtaining a vector for the first prediction unit.

11. The method of claim 9, wherein the coding unit comprises two or more prediction units, wherein each prediction unit has an associated vector, the method further comprising:

locating a first luma reference block using the vector associated with the first prediction unit;

locating a second luma reference block using the vector associated with the second prediction unit;

locating the first chroma reference block using the vector associated with the first prediction unit; and

locating the second chroma reference block using the vector associated with the first prediction unit.

12. The method of claim 9, wherein a chroma sampling format for the coding unit is 4:2:0, and wherein the vector determined from the plurality of vectors is associated with a bottom-right block of the coding unit.

13. The method of claim 12, wherein the second chroma block is a top-right block of the coding unit, the method further comprising:

locating a third reference block using the vector, wherein the third reference block is in the picture;

predicting a third chroma block based on the first reference block, wherein the third chroma block is a top-left block of the coding unit;

locating a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

predicting a fourth chroma block of the coding unit based on the second reference block, wherein the fourth chroma block is a bottom-left block of the coding unit.

14. The method of claim 9, wherein a chroma sampling format for the coding unit is 4:2:2, and wherein the first chroma block is a top-right block and the second chroma block is a top-left block of the coding unit.

15. The method of claim 14, further comprising:

determining a second vector for a third chroma block of the coding unit;

locating a third reference block using the second vector, wherein the third reference block is in the picture;

using the third reference block as a predictor for the third chroma block;

locating a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

using the fourth reference block as a predictor for a fourth chroma block of the coding unit.

16. The method of claim 9, wherein the coding unit comprises residual data.

17. A device for coding video data, the device comprising:
a memory configured to store video data;
a video coder comprising one or more processors configured to:
determine from a plurality of vectors associated with the coding unit a coding unit of a picture of the video data is coded using an intra block copy mode;
determine a vector for a first chroma block of the coding unit;
locate a first chroma reference block using the vector, wherein the first chroma reference block is in the picture;
predict the first chroma block based on the first chroma reference block;
using the same vector that was used to locate the first chroma reference block, locate a second chroma reference block for a second chroma block of the same coding unit, wherein the second chroma reference block is in the picture, and different from the first chroma reference block; and
predict a second chroma block of the coding unit based on the second chroma reference block.

18. The device of claim 17, wherein the coding unit comprises two or more prediction units and wherein the first chroma block corresponds to a first prediction unit and the second chroma block corresponds to a second prediction unit, and wherein determining the vector for the second chroma block of the coding unit comprises obtaining a vector for the first prediction unit.

19. The device of claim 17, wherein the coding unit comprises two or more prediction units, wherein each prediction unit has an associated vector, and wherein the one or more processors are further configured to:
locate a first luma reference block using the vector associated with the first prediction unit;
locate a second luma reference block using the vector associated with the second prediction unit;
locate the first chroma reference block using the vector associated with the first prediction unit; and

locate the second chroma reference block using the vector associated with the first prediction unit.

20. The device of claim 17, wherein a chroma sampling format for the coding unit is 4:2:0, and wherein the vector determined from the plurality of vectors is associated with a bottom-right block of the coding unit.

21. The device of claim 20, wherein the second chroma block is a top-right block of the coding unit, and wherein the one or more processors are further configured to:

locate a third reference block using the vector, wherein the third reference block is in the picture;

predict a third chroma block based on the first reference block, wherein the third chroma block is a top-left block of the coding unit;

locate a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

predict a fourth chroma block of the coding unit based on the second reference block, wherein the fourth chroma block is a bottom-left block of the coding unit.

22. The device of claim 17, wherein a chroma sampling format for the coding unit is 4:2:2, and wherein the first chroma block is a top-right block and the second chroma block is a top-left block of the coding unit.

23. The device of claim 22, and wherein the one or more processors are further configured to:

determine a second vector for a third chroma block of the coding unit;

locate a third reference block using the second vector, wherein the third reference block is in the picture;

use the third reference block as a predictor for the third chroma block;

locate a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

use the fourth reference block as a predictor for a fourth chroma block of the coding unit.

24. The device of claim 17, wherein the coding unit comprises residual data.

25. The device of claim 17, wherein the device comprises one of:
a microprocessor;
an integrated circuit (IC); and
a wireless communication device comprising the video encoder.

26. The device of claim 17, wherein the video coder comprises a video encoder.

27. The device of claim 17, wherein the video coder comprises a video decoder.

28. An apparatus for coding video data, the apparatus comprising:
means for determining from a plurality of vectors associated with the coding unit a coding unit of a picture of the video data is coded using an intra block copy mode;
means for determining a vector for a first chroma block of the coding unit;
means for using the same vector that was used to locate the first chroma reference block to locate a second chroma reference block for a second chroma block of the same coding unit, wherein the second chroma reference block is in the picture, and different from the first chroma reference block;
means for predicting the first chroma block based on the first chroma reference block;
means for locating a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; and
means for predicting a second chroma block of the coding unit based on the second chroma reference block.

29. The apparatus of claim 28, wherein the coding unit comprises two or more prediction units and wherein the first chroma block corresponds to a first prediction unit and the second chroma block corresponds to a second prediction unit, and wherein the means for determining the vector for the second chroma block of the coding unit comprises means for obtaining a vector for the first prediction unit.

30. The apparatus of claim 28, wherein the coding unit comprises two or more prediction units, wherein each prediction unit has an associated vector, the apparatus further comprising:

means for locating a first luma reference block using the vector associated with the first prediction unit;

means for locating a second luma reference block using the vector associated with the second prediction unit;

means for locating the first chroma reference block using the vector associated with the first prediction unit; and

means for locating the second chroma reference block using the vector associated with the first prediction unit.

31. The apparatus of claim 28, wherein a chroma sampling format for the coding unit is 4:2:0, and wherein the vector determined from the plurality of vectors is associated with a bottom-right block of the coding unit.

32. The apparatus of claim 31, wherein the second chroma block is a top-right block of the coding unit, the apparatus further comprising:

means for locating a third reference block using the vector, wherein the third reference block is in the picture;

means for predicting a third chroma block based on the first reference block, wherein the third chroma block is a top-left block of the coding unit;

means for locating a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

means for predicting a fourth chroma block of the coding unit based on the second reference block, wherein the fourth chroma block is a bottom-left block of the coding unit.

33. The apparatus of claim 28, wherein a chroma sampling format for the coding unit is 4:2:2, and wherein the first chroma block is a top-right block and the second chroma block is a top-left block of the coding unit.

34. The apparatus of claim 33, further comprising:

means for determining a second vector for a third chroma block of the coding unit;

means for locating a third reference block using the second vector, wherein the third reference block is in the picture;

means for using the third reference block as a predictor for the third chroma block;

means for locating a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

means for using the fourth reference block as a predictor for a fourth chroma block of the coding unit.

35. The apparatus of claim 28, wherein the coding unit comprises residual data.

36. A computer readable storage medium storing instructions that when executed by one or more processors cause the one or more processors to:

determine from a plurality of vectors associated with the coding unit a coding unit of a picture of the video data is coded using an intra block copy mode;

determine a vector for a first chroma block of the coding unit;

use the same vector that was used to locate the first chroma reference block to locate a second chroma reference block for a second chroma block of the same coding unit, wherein the second chroma reference block is in the picture, and different from the first chroma reference block;

predict the first chroma block based on the first chroma reference block;

locate a second chroma reference block using the vector, wherein the second chroma reference block is in the picture; and

predict a second chroma block of the coding unit based on the second chroma reference block.

37. The computer-readable storage medium of claim 36, wherein the coding unit comprises two or more prediction units and wherein the first chroma block corresponds to a first prediction unit and the second chroma block corresponds to a second prediction unit, and wherein determining the vector for the second chroma block of the coding unit comprises obtaining a vector for the first prediction unit.

38. The computer-readable storage medium of claim 36, wherein the coding unit comprises two or more prediction units, wherein each prediction unit has an associated vector, and wherein the computer-readable storage medium stores further instructions that when executed by the one or more processors cause the one or more processors to:

locate a first luma reference block using the vector associated with the first prediction unit;

locate a second luma reference block using the vector associated with the second prediction unit;

locate the first chroma reference block using the vector associated with the first prediction unit; and

locate the second chroma reference block using the vector associated with the first prediction unit.

39. The computer-readable storage medium of claim 36, wherein a chroma sampling format for the coding unit is 4:2:0, and wherein the vector determined from the plurality of vectors is associated with a bottom-right block of the coding unit.

40. The computer-readable storage medium of claim 39, wherein the second chroma block is a top-right block of the coding unit, and wherein the computer-readable storage medium stores further instructions that when executed by the one or more processors cause the one or more processors to:

locate a third reference block using the vector, wherein the third reference block is in the picture;

predict a third chroma block based on the first reference block, wherein the third chroma block is a top-left block of the coding unit;

locate a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

predict a fourth chroma block of the coding unit based on the second reference block, wherein the fourth chroma block is a bottom-left block of the coding unit.

41. The computer-readable storage medium of claim 36, wherein a chroma sampling format for the coding unit is 4:2:2, and wherein the first chroma block is a top-right block and the second chroma block is a top-left block of the coding unit.

42. The computer-readable storage medium of claim 41, storing further instructions that when executed by the one or more processors cause the one or more processors to:

determine a second vector for a third chroma block of the coding unit;

locate a third reference block using the second vector, wherein the third reference block is in the picture;

use the third reference block as a predictor for the third chroma block;

locate a fourth reference block using the vector, wherein the fourth reference block is in the picture; and

use the fourth reference block as a predictor for a fourth chroma block of the coding unit.

43. The computer-readable storage medium of claim 36, wherein the coding unit comprises residual data.

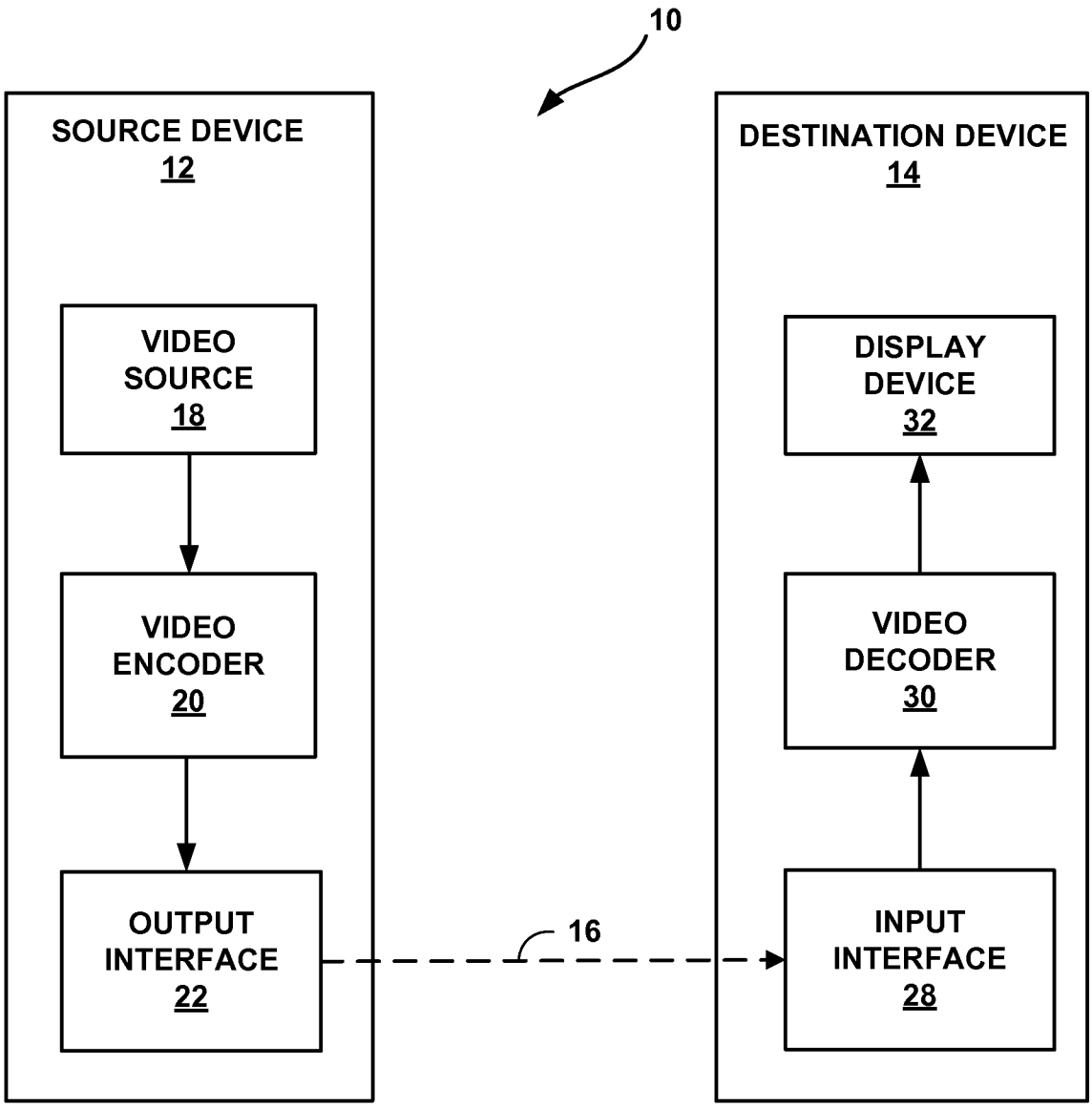
10
SOURCE DEVICE
12
VIDEO SOURCE
18
VIDEO ENCODER
20
OUTPUT INTERFACE
22
16
DESTINATION DEVICE
14
DISPLAY DEVICE
32
VIDEO DECODER
30
INPUT INTERFACE
28

FIG. 1

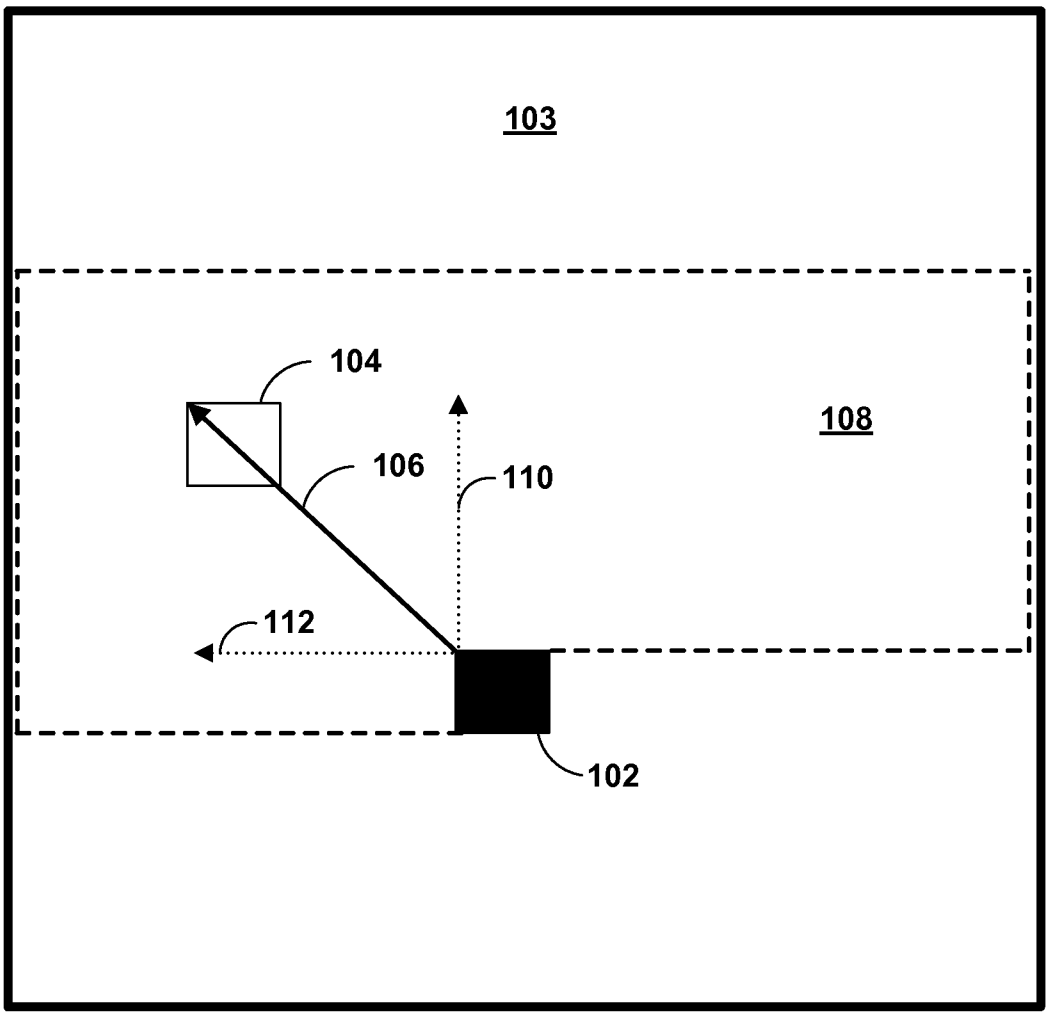

FIG. 2

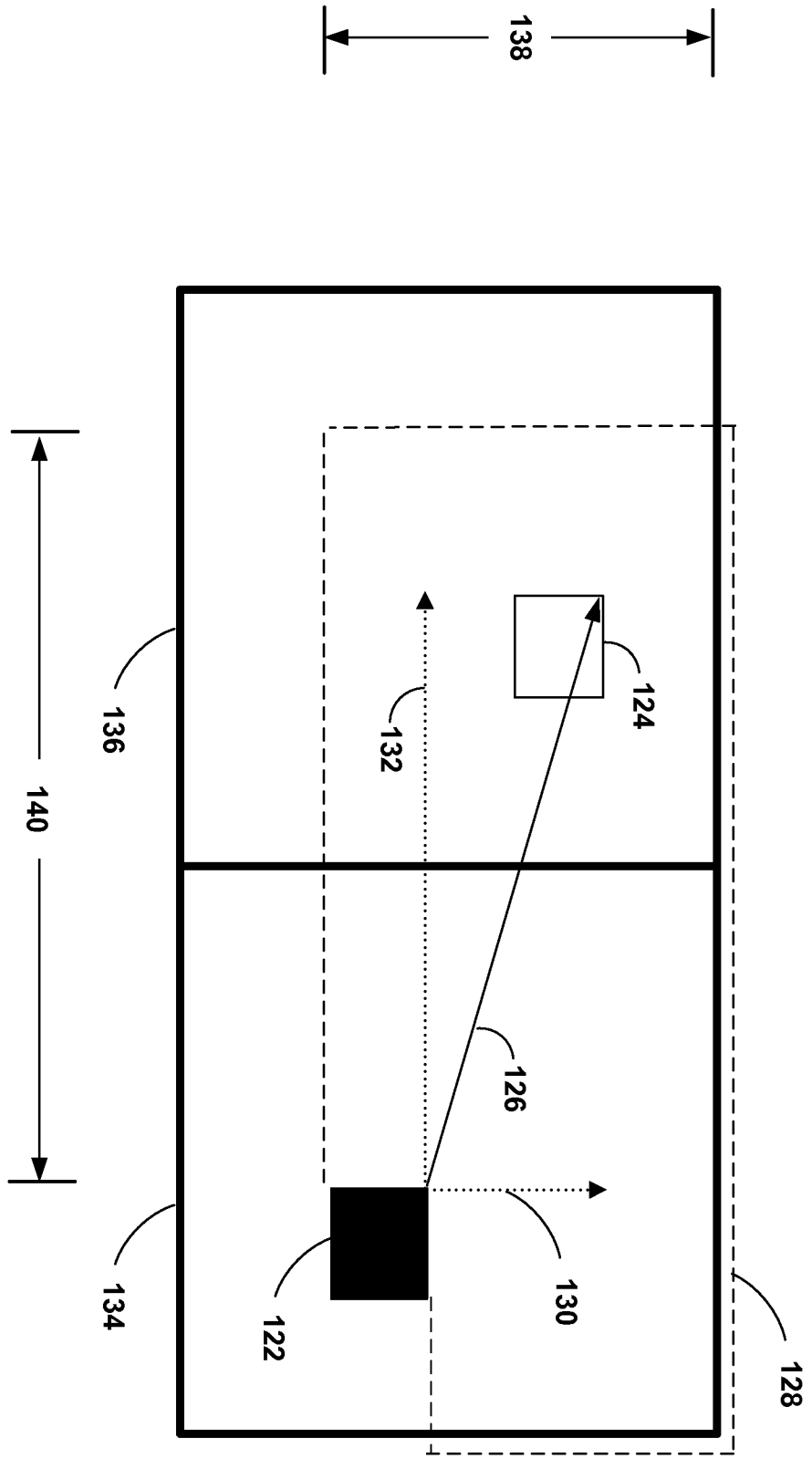

FIG. 3

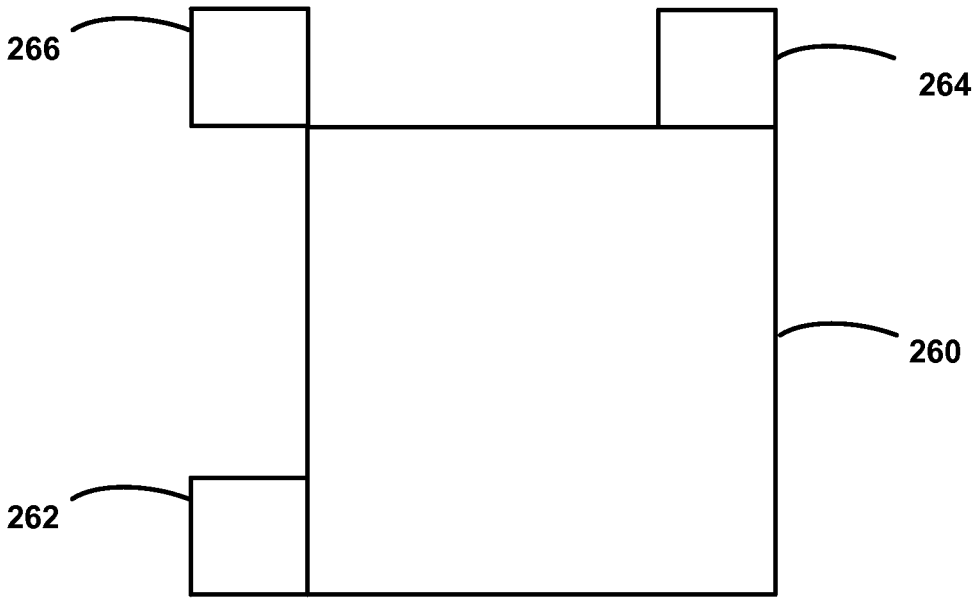

FIG. 4

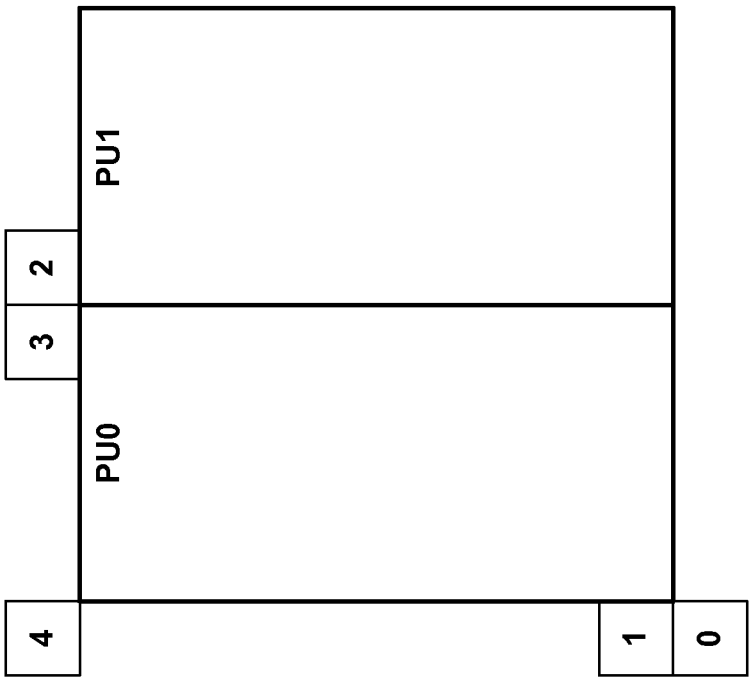

FIG. 5B

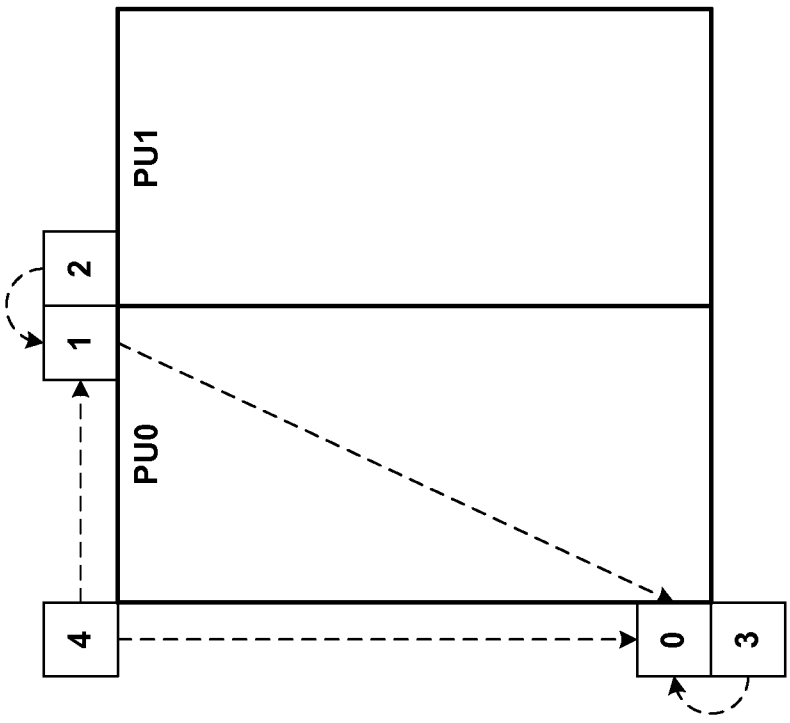

FIG. 5A

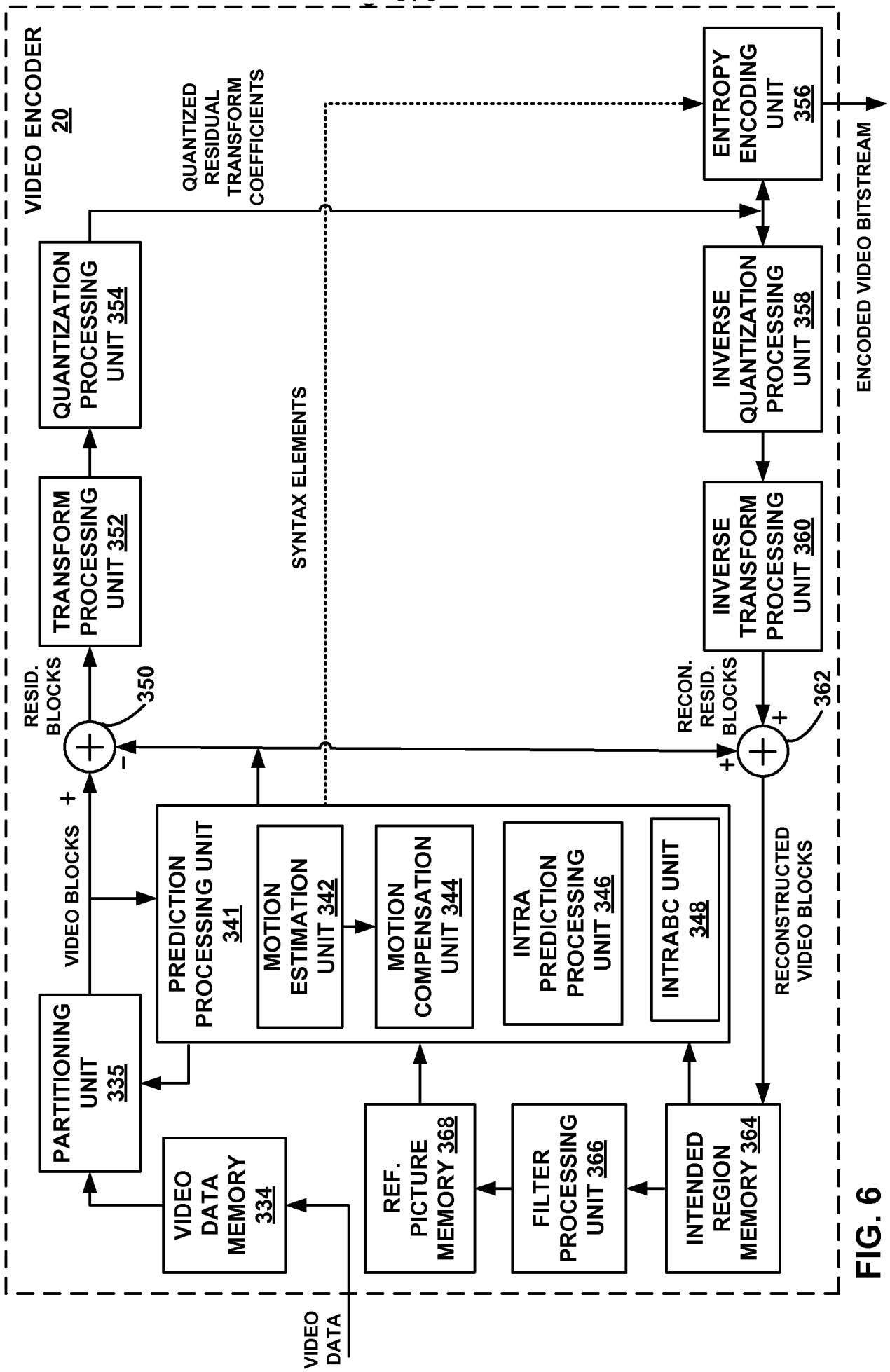
VIDEO ENCODER 20
VIDEO DATA
VIDEO DATA MEMORY 334
PARTITIONING UNIT 335
PREDICTION PROCESSING UNIT 341
MOTION ESTIMATION UNIT 342
MOTION COMPENSATION UNIT 344
INTRA PREDICTION PROCESSING UNIT 346
INTRABC UNIT 348
VIDEO BLOCKS
RESID. BLOCKS
350
TRANSFORM PROCESSING UNIT 352
QUANTIZATION PROCESSING UNIT 354
QUANTIZED RESIDUAL TRANSFORM COEFFICIENTS
SYNTAX ELEMENTS
ENTROPY ENCODING UNIT 356
ENCODED VIDEO BITSTREAM
INVERSE QUANTIZATION PROCESSING UNIT 358
INVERSE TRANSFORM PROCESSING UNIT 360
RECON. RESID. BLOCKS
362
RECONSTRUCTED VIDEO BLOCKS
REF. PICTURE MEMORY 368
FILTER PROCESSING UNIT 366
INTENDED REGION MEMORY 364

FIG. 6

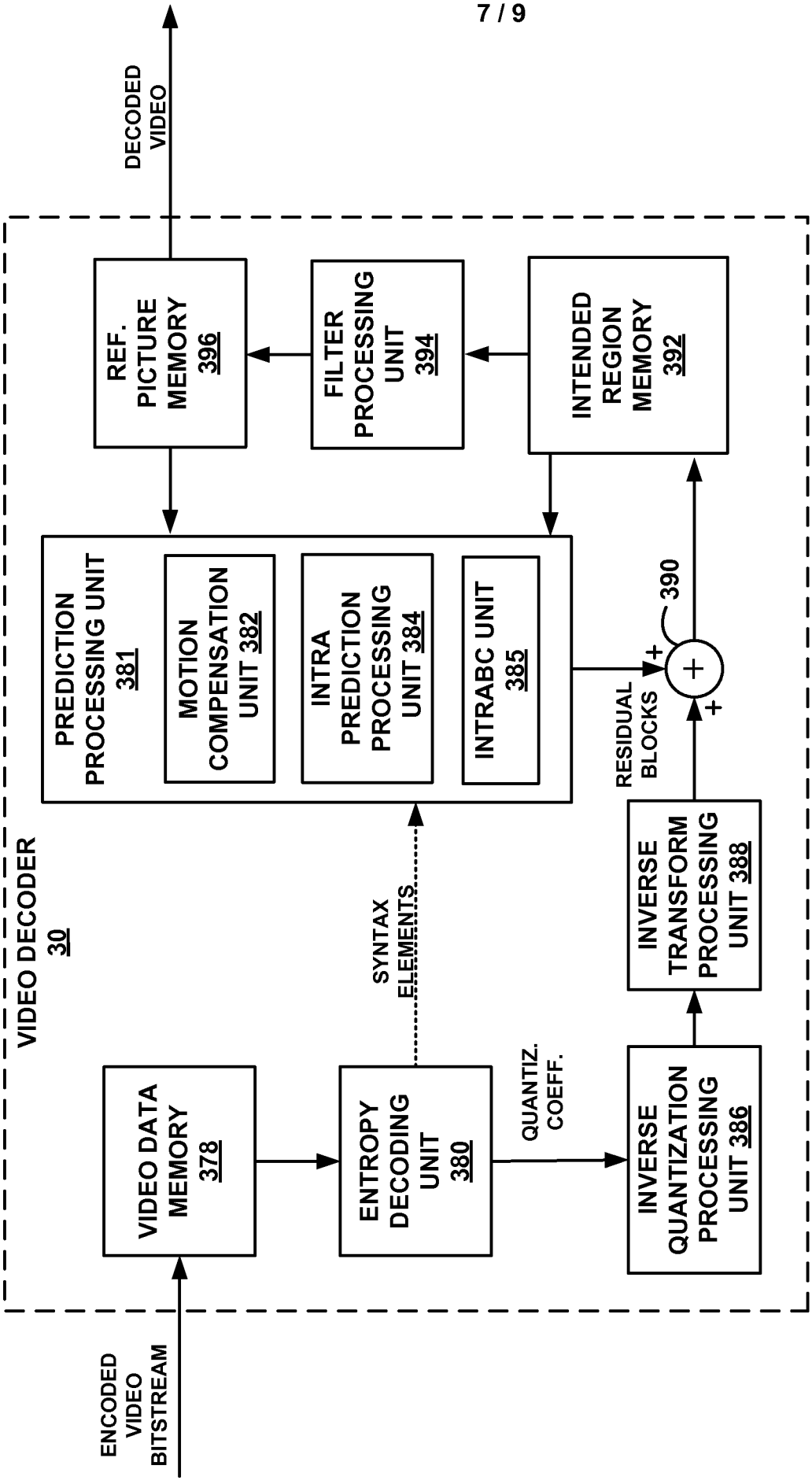
DECODED VIDEO
REF. PICTURE MEMORY 396
FILTER PROCESSING UNIT 394
INTENDED REGION MEMORY 392
VIDEO DECODER 30
PREDICTION PROCESSING UNIT 381
MOTION COMPENSATION UNIT 382
INTRA PREDICTION PROCESSING UNIT 384
INTRABC UNIT 385
390
RESIDUAL BLOCKS
SYNTAX ELEMENTS
INVERSE TRANSFORM PROCESSING UNIT 388
VIDEO DATA MEMORY 378
ENTROPY DECODING UNIT 380
QUANTIZ. COEFF.
INVERSE QUANTIZATION PROCESSING UNIT 386
ENCODED VIDEO BITSTREAM

FIG. 7

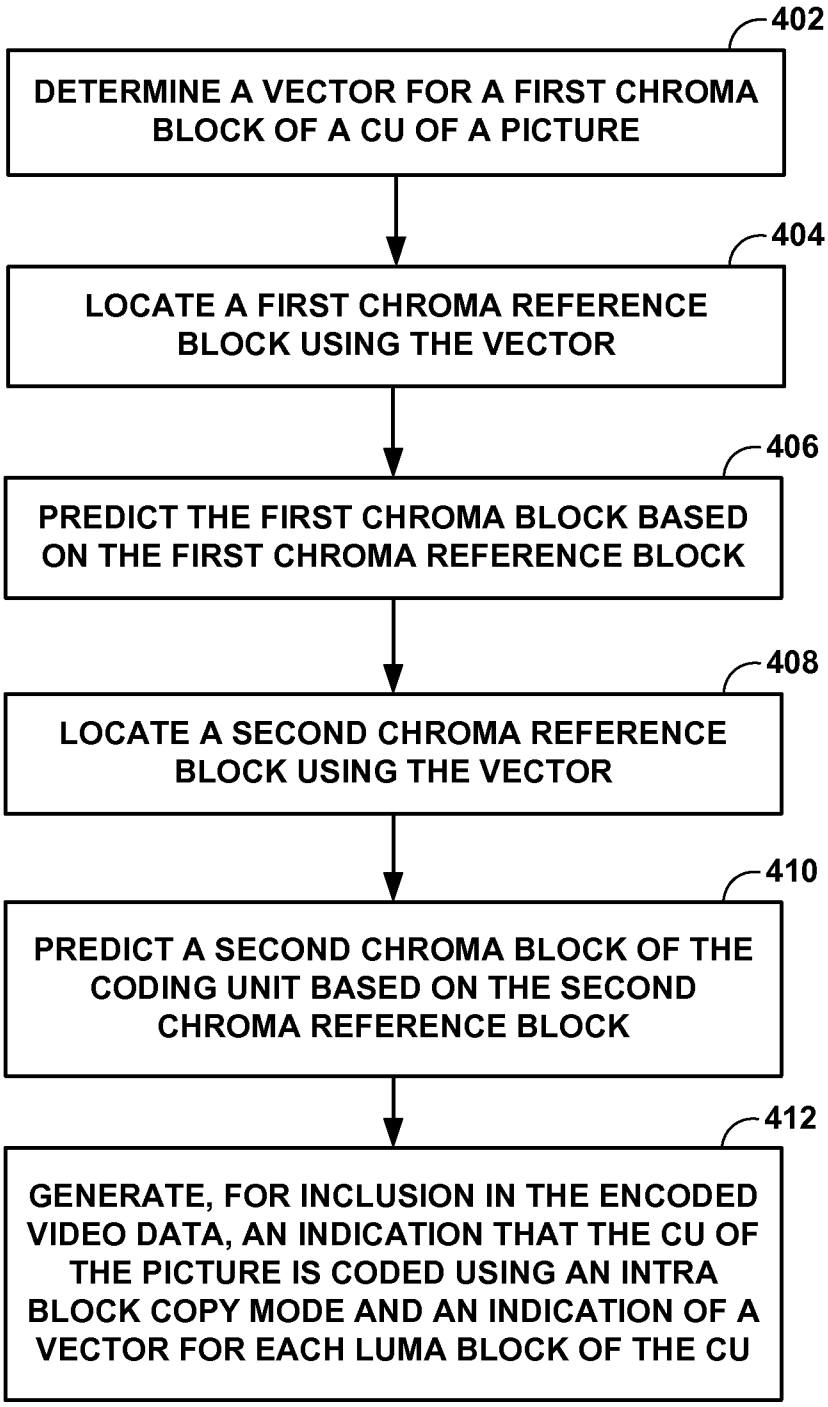
402
DETERMINE A VECTOR FOR A FIRST CHROMA BLOCK OF A CU OF A PICTURE
404
LOCATE A FIRST CHROMA REFERENCE BLOCK USING THE VECTOR
406
PREDICT THE FIRST CHROMA BLOCK BASED ON THE FIRST CHROMA REFERENCE BLOCK
408
LOCATE A SECOND CHROMA REFERENCE BLOCK USING THE VECTOR
410
PREDICT A SECOND CHROMA BLOCK OF THE CODING UNIT BASED ON THE SECOND CHROMA REFERENCE BLOCK
412
GENERATE, FOR INCLUSION IN THE ENCODED VIDEO DATA, AN INDICATION THAT THE CU OF THE PICTURE IS CODED USING AN INTRA BLOCK COPY MODE AND AN INDICATION OF A VECTOR FOR EACH LUMA BLOCK OF THE CU

FIG. 8

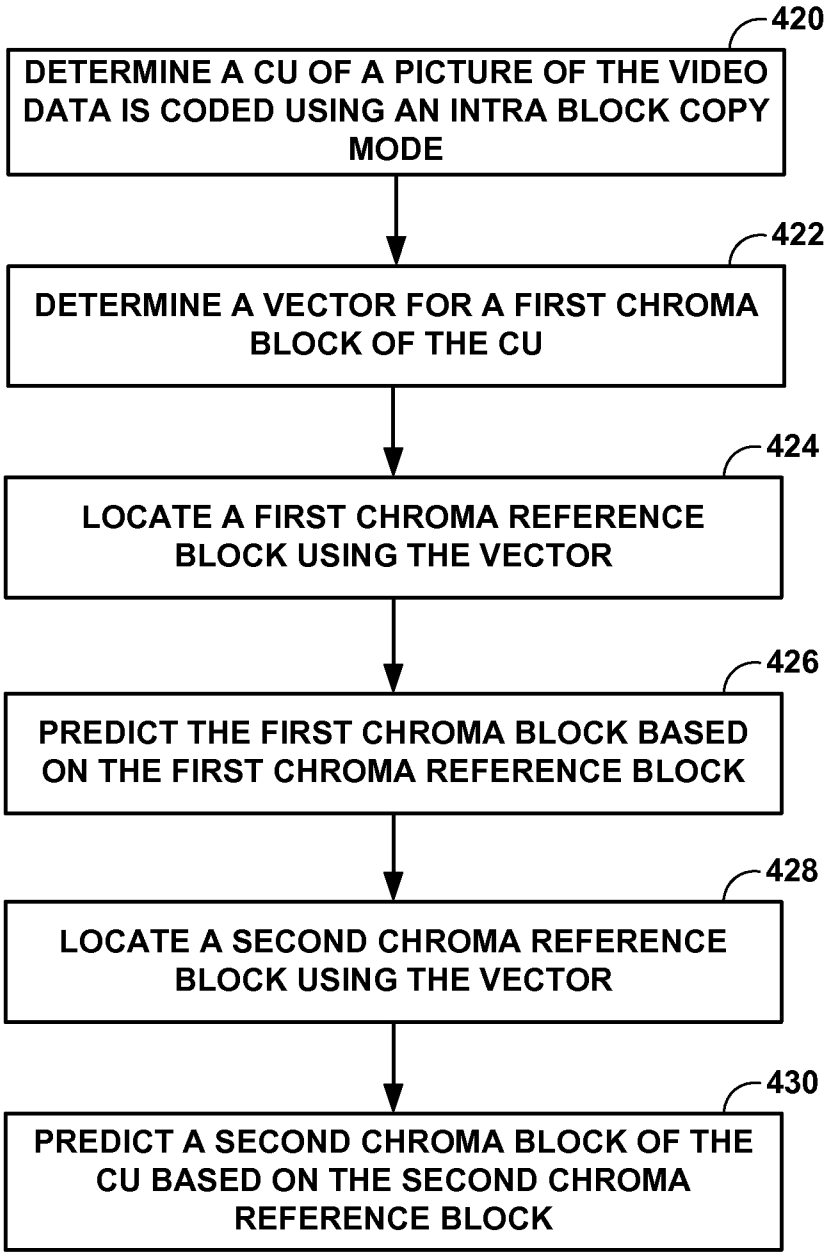
420
DETERMINE A CU OF A PICTURE OF THE VIDEO DATA IS CODED USING AN INTRA BLOCK COPY MODE
422
DETERMINE A VECTOR FOR A FIRST CHROMA BLOCK OF THE CU
424
LOCATE A FIRST CHROMA REFERENCE BLOCK USING THE VECTOR
426
PREDICT THE FIRST CHROMA BLOCK BASED ON THE FIRST CHROMA REFERENCE BLOCK
428
LOCATE A SECOND CHROMA REFERENCE BLOCK USING THE VECTOR
430
PREDICT A SECOND CHROMA BLOCK OF THE CU BASED ON THE SECOND CHROMA REFERENCE BLOCK

FIG. 9

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2015/010783

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N19/593 H04N19/186
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, INSPEC, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	ROSEWARNE C ET AL: "BoG report on Range Extensions", 16. JCT-VC MEETING; 9-1-2014 - 17-1-2014; SAN JOSE; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-P0288-v4, 14 January 2014 (2014-01-14), XP030115839, abstract page 9, last paragraph - page 10, last line	1-43

-/--

[X] Further documents are listed in the continuation of Box C.

[ ] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
2 April 2015	21/04/2015

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Kontopodis, D

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	NACCARI M ET AL: "HEVC Range Extensions Test Model 6 Encoder Description", 16. JCT-VC MEETING; 9-1-2014 - 17-1-2014; SAN JOSE; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-P1013, 23 February 2014 (2014-02-23), XP030115885, section 4.3	1-43

A	NACCARI M ET AL: "HEVC Range Extensions Test Model 5 Encoder Description", 15. JCT-VC MEETING; 23-10-2013 - 1-11-2013; GENEVA; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-O1013, 14 December 2013 (2013-12-14), XP030115465, section 4.3	1-43

A	PANG C ET AL: "Non-RCE3: 2-D MV Supported Intra Motion Compensation", 14. JCT-VC MEETING; 25-7-2013 - 2-8-2013; VIENNA; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/,, no. JCTVC-N0256, 16 July 2013 (2013-07-16) , XP030114776, abstract sections 1 and 2	1-43

A	SULLIVAN G J ET AL: "Meeting Report of 13th JCT-VC Meeting", 13. JCT-VC MEETING; 104. MPEG MEETING; 18-4-2013 - 26-4-2013; INCHEON; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/,, no. JCTVC-M1000, 27 July 2013 (2013-07-27) , XP030114425, page 99, last paragraph - page 100, line 21	1-43

	-/--	

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2015/010783

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	YU S-L; CHRYSAFIS C: "New Intra Prediction using Self-Frame MCP", 3. JVT MEETING; 60. MPEG MEETING; 06-05-2002 - 10-05-2002; FAIRFAX,US; (JOINT VIDEO TEAM OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16),, no. JVT-C151r1-L, 10 May 2002 (2002-05-10) , XP030005267, ISSN: 0000-0442 page 1 - page 2 -----	1-43