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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 January 2025 (16.01.2025)

WIPO | PCT

(10) International Publication Number
WO 2025/014786 A1

(51) International Patent Classification:
H04N 19/186 (2014.01) *H04N 19/117* (2014.01)
H04N 19/176 (2014.01) *H04N 19/149* (2014.01)
H04N 19/132 (2014.01)

(21) International Application Number:
PCT/US2024/036838

(22) International Filing Date:
04 July 2024 (04.07.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/525,472 07 July 2023 (07.07.2023) US

(71) Applicant: **BEIJING DAJIA INTERNET INFORMATION TECHNOLOGY CO., LTD.** [CN/CN]; Room 101, 8th Floor, Building 12, No. 16, Xierqi West Road, Haidian District, Beijing 100085 (CN).

(72) Inventor; and

(71) Applicant *(for MN only)*: **WANG, Xianglin** [US/US]; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US).

(72) Inventors: **YAN, Ning**; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US). **XIU, Xiaoyu**; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US). **KUO, Che-Wei**; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US). **JHU, Hong-Jheng**; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US). **CHEN, Wei**; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US). **MA, Changyue**; 8910 University Center Lane, Suite 400, San Diego, CA 92122 (US). **YU, Bing**; No. 29 Xi'erqi Middle Road, Qinghe Street, Haidian District, Beijing 100085 (CN).

(74) Agent: **SONG, Jiayu (Mandy)**; Bayes PLLC, 8260 Greensboro Drive, Suite 625, McLean, VA 22102 (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,

(54) Title: IMPROVED CROSS COMPONENT PREDICTION

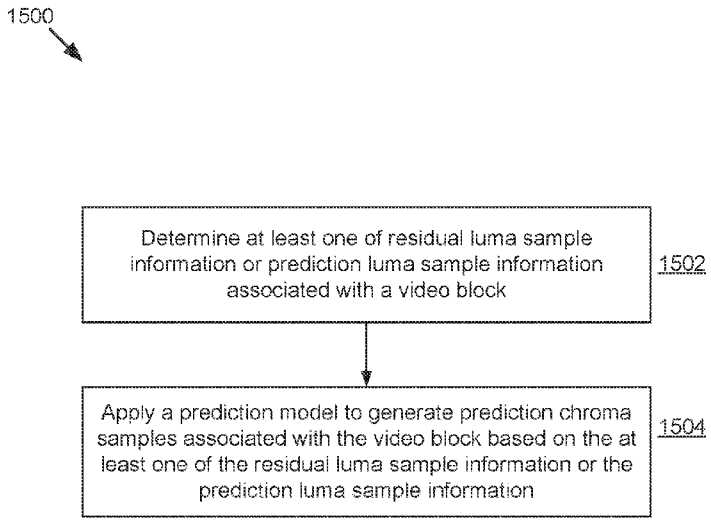


FIG. 15

(57) Abstract: A method for video encoding, a method for video decoding, and apparatuses thereof are provided. At least one of residual luma, sample information or prediction luma sample information associated with a video block from a video frame of a video is determined. A prediction model is applied to generate prediction chroma, samples associated with the video block based on the at least one of the residual luma, sample information or the prediction luma sample information.

WO 2025/014786 A1

[Continued on next page]

CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, 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, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, 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, ME, 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))*

IMPROVED CROSS COMPONENT PREDICTION

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application is based upon and claims priority to U.S. Provisional Application No. 63/525,472 filed July 7, 2023, the content of which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] This application is related to video coding and compression. More specifically, this application relates to video processing apparatuses and methods for cross component prediction.

BACKGROUND

[0003] Digital video is supported by a variety of electronic devices, such as digital televisions, laptop or desktop computers, tablet computers, digital cameras, digital recording devices, digital media players, video gaming consoles, smart phones, video teleconferencing devices, video streaming devices, etc. The electronic devices transmit and receive or otherwise communicate digital video data across a communication network, and/or store the digital video data on a storage device. Due to a limited bandwidth capacity of the communication network and limited memory resources of the storage device, video coding may be used to compress the video data according to one or more video coding standards before it is communicated or stored. For example, video coding standards include Versatile Video Coding (VVC), Joint Exploration test Model (JEM), High-Efficiency Video Coding (HEVC/H.265), Advanced Video Coding (AVC/H.264), Moving Picture Expert Group (MPEG) coding, or the like. Video coding generally utilizes prediction methods (e.g., inter-prediction, intra-prediction, or the like) that take advantage of redundancy inherent in the video data. Video coding aims to compress video data into a form that uses a lower bit rate, while avoiding or minimizing degradations to video quality.

SUMMARY

[0004] Implementations of the present disclosure provide a method for video decoding. The method may include determining, by a decoder, at least one of residual luma sample information or prediction luma sample information associated with a video block from a video frame of a video. The method may also include applying, by the decoder, a prediction model to generate prediction chroma samples associated with the video block based on the at least

one of the residual luma sample information or the prediction luma sample information.

[0005] Implementations of the present disclosure provide a method for video encoding. The method may include determining, by an encoder, at least one of residual luma sample information or prediction luma sample information associated with a video block from a video frame of a video. The method may also include applying, by the encoder, a prediction model to generate prediction chroma samples associated with the video block based on the at least one of the residual luma sample information or the prediction luma sample information.

[0006] Implementations of the present disclosure also provide a method at a decoder side. The method includes receiving a bitstream by a decoder. The bitstream is decoded by performing the method of video decoding disclosed herein.

[0007] Implementations of the present disclosure also provide a method at an encoder side. The method includes storing a bitstream by an encoder. The bitstream is generated by performing the method of video encoding disclosed herein.

[0008] Implementations of the present disclosure also provide an apparatus for video decoding. The apparatus may include a memory configured to store a bitstream and a processor coupled to the memory. The processor may be configured to perform a method for video decoding disclosed herein to decode the bitstream or a method at a decoder side disclosed herein to receive the bitstream.

[0009] Implementations of the present disclosure also provide an apparatus for video encoding. The apparatus may include a memory configured to store a bitstream and a processor coupled to the memory. The processor may be configured to perform a method for video encoding disclosed herein to generate the bitstream or a method at an encoder side disclosed herein to store the bitstream.

[0010] Implementations of the present disclosure also provide a non-transitory computer-readable storage medium having stored therein a bitstream, where the bitstream is decoded by a method for video decoding disclosed herein or the bitstream is received by a method at a decoder side disclosed herein.

[0011] Implementations of the present disclosure also provide a non-transitory computer-readable storage medium having stored therein a bitstream, where the bitstream is generated by a method for video encoding disclosed herein or the bitstream is stored by a method at an encoder side disclosed herein.

[0012] It is to be understood that both the foregoing general description and the following detailed description are examples only and are not restrictive of the present disclosure.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate examples consistent with the present disclosure and, together with the description, serve to explain the principles of the disclosure.

[0014] FIG. 1 is a block diagram illustrating an exemplary system for encoding and decoding video blocks in accordance with some implementations of the present disclosure.

[0015] FIG. 2 is a block diagram illustrating an exemplary video encoder in accordance with some implementations of the present disclosure.

[0016] FIG. 3 is a block diagram illustrating an exemplary video decoder in accordance with some implementations of the present disclosure.

[0017] FIGS. 4A-4E are block diagrams illustrating how a frame is recursively partitioned into multiple video blocks of different sizes and shapes in accordance with some implementations of the present disclosure.

[0018] FIG. 5 shows locations of left samples and above samples of a coding unit (CU) involved in a Cross Component Linear Model (CCLM) mode in accordance with some implementations of the present disclosure.

[0019] FIG. 6 shows a process of slope adjustment in the CCLM mode in accordance with some implementations of the present disclosure.

[0020] FIG. 7 shows spatial locations of luma samples applied in a convolutional filter in accordance with some implementations of the present disclosure.

[0021] FIG. 8 shows a reference area used to derive filter coefficients of a convolutional filter in accordance with some implementations of the present disclosure.

[0022] FIG. 9 shows a reference area with paddings used to derive filter coefficients of a Convolutional Cross Component Model (CCCM) in accordance with some implementations of the present disclosure.

[0023] FIG. 10 shows four gradient filters for a Gradient Linear Model (GLM) in accordance with some implementations of the present disclosure.

[0024] FIG. 11 shows luma samples for the CCCM without down-sampling in accordance with some implementations of the present disclosure.

[0025] FIG. 12 shows a diagram of gradient calculation in a Gradient and Location based Convolutional Cross Component Model (GL-CCCM) in accordance with some implementations of the present disclosure.

[0026] FIG. 13 illustrates a diagram of various down-sampling filters used in the CCCM

in accordance with some implementations of the present disclosure.

[0027] FIG. 14 shows a low-pass filter that is applied to filter prediction samples generated by Multi-Model CCLM (MM-CCLM) and/or Multi-Model CCCM (MM-CCCM) in accordance with some implementations of the present disclosure.

[0028] FIG. 15 is a flow chart of an exemplary method for video coding in accordance with some implementations of the present disclosure.

[0029] FIG. 16 illustrates residual luma samples without down-sampling in accordance with some implementations of the present disclosure.

[0030] FIG. 17 illustrates prediction luma samples without down-sampling in accordance with some implementations of the present disclosure.

[0031] FIG. 18 is a diagram illustrating a computing environment coupled with a user interface, according to some implementations of the present disclosure.

DETAILED DESCRIPTION

[0032] Reference will now be made in detail to specific implementations, examples of which are illustrated in the accompanying drawings. In the following detailed description, numerous non-limiting specific details are set forth in order to assist in understanding the subject matter presented herein. But it will be apparent to one of ordinary skill in the art that various alternatives may be used without departing from the scope of claims and the subject matter may be practiced without these specific details. For example, it will be apparent to one of ordinary skill in the art that the subject matter presented herein can be implemented on many types of electronic devices with digital video capabilities.

[0033] It should be understood that the terms “first,” “second,” and the like used in the description, claims of the present disclosure, and the accompanying drawings are used to distinguish objects, and not used to describe any specific order or sequence. It should be understood that the data used in this way may be interchanged under an appropriate condition, such that the embodiments of the present disclosure described herein may be implemented in orders besides those shown in the accompanying drawings or described in the present disclosure.

[0034] FIG. 1 is a block diagram illustrating an exemplary system 10 for encoding and decoding video blocks in parallel in accordance with some implementations of the present disclosure. As shown in FIG. 1, the system 10 includes a source device 12 that generates and encodes video data to be decoded at a later time by a destination device 14. The source device 12 and the destination device 14 may comprise any of a wide variety of electronic devices,

including desktop or laptop computers, tablet computers, smart phones, set-top boxes, digital televisions, cameras, display devices, digital media players, video gaming consoles, video streaming device, or the like. In some implementations, the source device 12 and the destination device 14 are equipped with wireless communication capabilities.

[0035] In some implementations, the destination device 14 may receive the encoded video data to be decoded via a link 16. The link 16 may comprise any type of communication medium or device capable of moving the encoded video data from the source device 12 to the destination device 14. In one example, the link 16 may comprise a communication medium to enable the source device 12 to transmit the encoded video data directly to the 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 the 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 the source device 12 to the destination device 14.

[0036] In some other implementations, the encoded video data may be transmitted from an output interface 22 to a storage device 32. Subsequently, the encoded video data in the storage device 32 may be accessed by the destination device 14 via an input interface 28. The storage device 32 may include any of a variety of distributed or locally accessed data storage media such as a hard drive, Blu-ray discs, Digital Versatile Disks (DVDs), Compact Disc Read-Only Memories (CD-ROMs), flash memory, volatile or non-volatile memory, or any other suitable digital storage media for storing the encoded video data. In a further example, the storage device 32 may correspond to a file server or another intermediate storage device that may hold the encoded video data generated by the source device 12. The destination device 14 may access the stored video data from the storage device 32 via streaming or downloading. The file server may be any type of computer capable of storing the encoded video data and transmitting the encoded video data to the destination device 14. Exemplary file servers include a web server (e.g., for a website), a File Transfer Protocol (FTP) server, Network Attached Storage (NAS) devices, or a local disk drive. The destination device 14 may access the encoded video data through any standard data connection, including a wireless channel (e.g., a Wireless Fidelity (Wi-Fi) connection), a wired connection (e.g., Digital Subscriber Line (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 the encoded video data from the storage device 32 may be a streaming transmission, a download transmission, or a combination of both.

[0037] As shown in FIG. 1, the source device 12 includes a video source 18, a video encoder 20 and the output interface 22. The video source 18 may include a source such as a video capturing device, e.g., a video camera, a video archive containing previously captured video, a video feeding interface to receive video from a video content provider, and/or a computer graphics system for generating computer graphics data as the source video, or a combination of such sources. As one example, if the video source 18 is a video camera of a security surveillance system, the source device 12 and the destination device 14 may form camera phones or video phones. However, the implementations described in the present application may be applicable to video coding in general, and may be applied to wireless and/or wired applications.

[0038] The captured, pre-captured, or computer-generated video may be encoded by the video encoder 20. The encoded video data may be transmitted directly to the destination device 14 via the output interface 22 of the source device 12. The encoded video data may also (or alternatively) be stored onto the storage device 32 for later access by the destination device 14 or other devices, for decoding and/or playback. The output interface 22 may further include a modem and/or a transmitter.

[0039] The destination device 14 includes the input interface 28, a video decoder 30, and a display device 34. The input interface 28 may include a receiver and/or a modem and receive the encoded video data over the link 16. The encoded video data communicated over the link 16, or provided on the storage device 32, may include a variety of syntax elements generated by the video encoder 20 for use by the video decoder 30 in decoding the video data. Such syntax elements may be included within the encoded video data transmitted on a communication medium, stored on a storage medium, or stored on a file server.

[0040] In some implementations, the destination device 14 may include the display device 34, which can be an integrated display device and an external display device that is configured to communicate with the destination device 14. The display device 34 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a Liquid Crystal Display (LCD), a plasma display, an Organic Light Emitting Diode (OLED) display, or another type of display device.

[0041] The video encoder 20 and the video decoder 30 may operate according to proprietary or industry standards, such as VVC, HEVC, MPEG-4, Part 10, AVC, or extensions of such standards. It should be understood that the present application is not limited to a specific

video encoding/decoding standard and may be applicable to other video encoding/decoding standards. It is generally contemplated that the video encoder 20 of the source device 12 may be configured to encode video data according to any of these current or future standards. Similarly, it is also generally contemplated that the video decoder 30 of the destination device 14 may be configured to decode video data according to any of these current or future standards.

[0042] The video encoder 20 and the video decoder 30 each may be implemented as any of a variety of suitable encoder and/or decoder 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 implemented partially in software, an electronic 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 video encoding/decoding operations disclosed in the present disclosure. Each of the video encoder 20 and the 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.

[0043] In some implementations, at least a part of components of the source device 12 (for example, the video source 18, the video encoder 20 or components included in the video encoder 20 as described below with reference to Fig. 2, and the output interface 22) and/or at least a part of components of the destination device 14 (for example, the input interface 28, the video decoder 30 or components included in the video decoder 30 as described below with reference to Fig. 3, and the display device 34) may operate in a cloud computing service network which may provide software, platforms, and/or infrastructure, such as Software as a Service (SaaS), Platform as a Service (PaaS), or Infrastructure as a Service (IaaS). In some implementations, one or more components in the source device 12 and/or the destination device 14 which are not included in the cloud computing service network may be provided in one or more client devices, and the one or more client devices may communicate with server computers in the cloud computing service network through a wireless communication network (for example, a cellular communication network, a short-range wireless communication network, or a global navigation satellite system (GNSS) communication network) or a wired communication network (e.g., a local area network (LAN) communication network or a power line communication (PLC) network). In an embodiment, at least a part of operations described herein may be implemented as cloud-based services provided by one or more server computers which are implemented by the at least a part of the components of the source device 12 and/or

the at least a part of the components of the destination device 14 in the cloud computing service network; and one or more other operations described herein may be implemented by the one or more client devices. In some implementations, the cloud computing service network may be a private cloud, a public cloud, or a hybrid cloud. The terms such as "cloud," "cloud computing," "cloud-based" etc. herein may be used interchangeably as appropriate without departing from the scope of the present disclosure. It should be understood that the present disclosure is not limited to being implemented in the cloud computing service network described above. Instead, the present disclosure may also be implemented in any other type of computing environments currently known or developed in the future.

[0044] FIG. 2 is a block diagram illustrating an exemplary video encoder 20 in accordance with some implementations described in the present application. The video encoder 20 may perform intra and inter predictive coding of video blocks within video frames. Intra predictive coding relies on spatial prediction to reduce or remove spatial redundancy in video data within a given video frame or picture. Inter predictive coding relies on temporal prediction to reduce or remove temporal redundancy in video data within adjacent video frames or pictures of a video sequence. It should be noted that the term "frame" may be used as synonyms for the term "image" or "picture" in the field of video coding.

[0045] As shown in FIG. 2, the video encoder 20 includes a video data memory 40, a prediction processing unit 41, a Decoded Picture Buffer (DPB) 64, a summer 50, a transform processing unit 52, a quantization unit 54, and an entropy encoding unit 56. The prediction processing unit 41 further includes a motion estimation unit 42, a motion compensation unit 44, a partition unit 45, an intra prediction processing unit 46, and an intra Block Copy (BC) unit 48. In some implementations, the video encoder 20 also includes an inverse quantization unit 58, an inverse transform processing unit 60, and a summer 62 for video block reconstruction. An in-loop filter 63, such as a deblocking filter, may be positioned between the summer 62 and the DPB 64 to filter block boundaries to remove blockiness artifacts from reconstructed video. Another in-loop filter, such as Sample Adaptive Offset (SAO) filter, Cross Component Sample Adaptive Offset (CCSAO) filter and/or Adaptive in-Loop Filter (ALF), may also be used in addition to the deblocking filter to filter an output of the summer 62. It should be understood that for the CCSAO technique, the present application is not limited to the embodiments described herein, and instead, the application may be applied to a situation where an offset is selected for any of a luma component, a Cb chroma component and a Cr chroma component according to any other of the luma component, the Cb chroma component and the Cr chroma component to modify said any component based on the selected offset.

Further, it should also be understood that a first component mentioned herein may be any of the luma component, the Cb chroma component and the Cr chroma component, a second component mentioned herein may be any other of the luma component, the Cb chroma component and the Cr chroma component, and a third component mentioned herein may be a remaining one of the luma component, the Cb chroma component and the Cr chroma component. In some examples, the in-loop filters may be omitted, and the decoded video block may be directly provided by the summer 62 to the DPB 64. The video encoder 20 may take the form of a fixed or programmable hardware unit or may be divided among one or more of the illustrated fixed or programmable hardware units.

[0046] The video data memory 40 may store video data to be encoded by the components of the video encoder 20. The video data in the video data memory 40 may be obtained, for example, from the video source 18 as shown in FIG. 1. The DPB 64 is a buffer that stores reference video data (for example, reference frames or pictures) for use in encoding video data by the video encoder 20 (e.g., in intra or inter predictive coding modes). The video data memory 40 and the DPB 64 may be formed by any of a variety of memory devices. In various examples, the video data memory 40 may be on-chip with other components of the video encoder 20, or off-chip relative to those components.

[0047] As shown in FIG. 2, after receiving the video data, the partition unit 45 within the prediction processing unit 41 partitions the video data into video blocks. This partitioning may also include partitioning a video frame into slices, tiles (for example, sets of video blocks), or other larger Coding Units (CUs) according to predefined splitting structures such as a Quad-Tree (QT) structure associated with the video data. The video frame is or may be regarded as a two-dimensional array or matrix of samples with sample values. A sample in the array may also be referred to as a pixel or a pel. A number of samples in horizontal and vertical directions (or axes) of the array or picture define a size and/or a resolution of the video frame. The video frame may be divided into multiple video blocks by, for example, using QT partitioning. The video block again is or may be regarded as a two-dimensional array or matrix of samples with sample values, although of smaller dimension than the video frame. A number of samples in horizontal and vertical directions (or axes) of the video block define a size of the video block. The video block may further be partitioned into one or more block partitions or sub-blocks (which may form again blocks) by, for example, iteratively using QT partitioning, Binary-Tree (BT) partitioning or Triple-Tree (TT) partitioning or any combination thereof. It should be noted that the term “block” or “video block” as used herein may be a portion, in particular a rectangular (square or non- square) portion, of a frame or a picture. With reference, for example,

to HEVC and VVC, the block or video block may be or correspond to a Coding Tree Unit (CTU), a CU, a Prediction Unit (PU) or a Transform Unit (TU) and/or may be or correspond to a corresponding block, e.g. a Coding Tree Block (CTB), a Coding Block (CB), a Prediction Block (PB) or a Transform Block (TB) and/or to a sub-block.

[0048] The prediction processing unit 41 may select one of a plurality of possible predictive coding modes, such as one of a plurality of intra predictive coding modes or one of a plurality of inter predictive coding modes, for the current video block based on error results (e.g., coding rate and the level of distortion). The prediction processing unit 41 may provide the resulting intra or inter prediction coded block to the summer 50 to generate a residual block and to the summer 62 to reconstruct the encoded block for use as part of a reference frame subsequently. The prediction processing unit 41 also provides syntax elements, such as motion vectors, intra-mode indicators, partition information, and other such syntax information, to the entropy encoding unit 56.

[0049] In order to select an appropriate intra predictive coding mode for the current video block, the intra prediction processing unit 46 within the prediction processing unit 41 may perform intra predictive coding of the current video block relative to one or more neighbor blocks in the same frame as the current block to be coded to provide spatial prediction. The motion estimation unit 42 and the motion compensation unit 44 within the prediction processing unit 41 perform inter predictive coding of the current video block relative to one or more predictive blocks in one or more reference frames to provide temporal prediction. The video encoder 20 may perform multiple coding passes, e.g., to select an appropriate coding mode for each block of video data.

[0050] In some implementations, the motion estimation unit 42 determines the inter prediction mode for a current video frame by generating a motion vector, which indicates the displacement of a video block within the current video frame relative to a predictive block within a reference video frame, according to a predetermined pattern within a sequence of video frames. Motion estimation, performed by the motion estimation unit 42, is the process of generating motion vectors, which estimate motion for video blocks. A motion vector, for example, may indicate the displacement of a video block within a current video frame or picture relative to a predictive block within a reference frame relative to the current block being coded within the current frame. The predetermined pattern may designate video frames in the sequence as P frames or B frames. The intra BC unit 48 may determine vectors, e.g., block vectors, for intra BC coding in a manner similar to the determination of motion vectors by the motion estimation unit 42 for inter prediction, or may utilize the motion estimation unit 42 to

determine the block vector.

[0051] A predictive block for the video block may be or may correspond to a block or a reference block of a reference frame that is deemed as closely matching 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 implementations, the video encoder 20 may calculate values for sub-integer pixel positions of reference frames stored in the DPB 64. For example, the video encoder 20 may interpolate values of one-quarter pixel positions, one-eighth pixel positions, or other fractional pixel positions of the reference frame. Therefore, the motion estimation unit 42 may perform a motion search relative to the full pixel positions and fractional pixel positions and output a motion vector with fractional pixel precision.

[0052] The motion estimation unit 42 calculates a motion vector for a video block in an inter prediction coded frame by comparing the position of the video block to the position of a predictive block of a reference frame selected from a first reference frame list (List 0) or a second reference frame list (List 1), each of which identifies one or more reference frames stored in the DPB 64. The motion estimation unit 42 sends the calculated motion vector to the motion compensation unit 44 and then to the entropy encoding unit 56.

[0053] Motion compensation, performed by the motion compensation unit 44, may involve fetching or generating the predictive block based on the motion vector determined by the motion estimation unit 42. Upon receiving the motion vector for the current video block, the motion compensation unit 44 may locate a predictive block to which the motion vector points in one of the reference frame lists, retrieve the predictive block from the DPB 64, and forward the predictive block to the summer 50. The summer 50 then forms a residual video block of pixel difference values by subtracting pixel values of the predictive block provided by the motion compensation unit 44 from the pixel values of the current video block being coded. The pixel difference values forming the residual video block may include luma or chroma component differences or both. The motion compensation unit 44 may also generate syntax elements associated with the video blocks of a video frame for use by the video decoder 30 in decoding the video blocks of the video frame. The syntax elements may include, for example, syntax elements defining the motion vector used to identify the predictive block, any flags indicating the prediction mode, or any other syntax information described herein. Note that the motion estimation unit 42 and the motion compensation unit 44 may be highly integrated, but are illustrated separately for conceptual purposes.

[0054] In some implementations, the intra BC unit 48 may generate vectors and fetch

predictive blocks in a manner similar to that described above in connection with the motion estimation unit 42 and the motion compensation unit 44, but with the predictive blocks being in the same frame as the current block being coded and with the vectors being referred to as block vectors as opposed to motion vectors. In particular, the intra BC unit 48 may determine an intra-prediction mode to use to encode a current block. In some examples, the intra BC unit 48 may encode a current block using various intra-prediction modes, e.g., during separate encoding passes, and test their performance through rate-distortion analysis. Next, the intra BC unit 48 may select, among the various tested intra-prediction modes, an appropriate intra-prediction mode to use and generate an intra-mode indicator accordingly. For example, the intra BC unit 48 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 as the appropriate intra-prediction mode to use. 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 bitrate (i.e., a number of bits) used to produce the encoded block. Intra BC unit 48 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.

[0055] In other examples, the intra BC unit 48 may use the motion estimation unit 42 and the motion compensation unit 44, in whole or in part, to perform such functions for Intra BC prediction according to the implementations described herein. In either case, for Intra block copy, a predictive block may be a block that is deemed as closely matching the block to be coded, in terms of pixel difference, which may be determined by SAD, SSD, or other difference metrics, and identification of the predictive block may include calculation of values for sub-integer pixel positions.

[0056] Whether the predictive block is from the same frame according to intra prediction, or a different frame according to inter prediction, the 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 forming the residual video block may include both luma and chroma component differences.

[0057] The intra prediction processing unit 46 may intra-predict a current video block, as an alternative to the inter-prediction performed by the motion estimation unit 42 and the motion compensation unit 44, or the intra block copy prediction performed by the intra BC unit 48, as described above. In particular, the intra prediction processing unit 46 may determine an intra prediction mode to use to encode a current block. To do so, the intra prediction processing unit

46 may encode a current block using various intra prediction modes, e.g., during separate encoding passes, and the intra prediction processing unit 46 (or a mode selection unit, in some examples) may select an appropriate intra prediction mode to use from the tested intra prediction modes. The intra prediction processing unit 46 may provide information indicative of the selected intra-prediction mode for the block to the entropy encoding unit 56. The entropy encoding unit 56 may encode the information indicating the selected intra-prediction mode in the bitstream.

[0058] After the prediction processing unit 41 determines the predictive block for the current video block via either inter prediction or intra prediction, the summer 50 forms a residual video block by subtracting the predictive block from the current video block. The residual video data in the residual block may be included in one or more TUs and is provided to the transform processing unit 52. The transform processing unit 52 transforms the residual video data into residual transform coefficients using a transform, such as a Discrete Cosine Transform (DCT) or a conceptually similar transform.

[0059] The transform processing unit 52 may send the resulting transform coefficients to the quantization unit 54. The quantization unit 54 quantizes the transform coefficients to further reduce the bit rate. The quantization process may also 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, the quantization unit 54 may then perform a scan of a matrix including the quantized transform coefficients. Alternatively, the entropy encoding unit 56 may perform the scan.

[0060] Following quantization, the entropy encoding unit 56 entropy encodes the quantized transform coefficients into a video bitstream using, e.g., 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. The encoded bitstream may then be transmitted to the video decoder 30 as shown in FIG. 1, or archived in the storage device 32 as shown in FIG. 1 for later transmission to or retrieval by the video decoder 30. The entropy encoding unit 56 may also entropy encode the motion vectors and the other syntax elements for the current video frame being coded.

[0061] The inverse quantization unit 58 and the inverse transform processing unit 60 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual video block in the pixel domain for generating a reference block for prediction of other video blocks. As noted above, the motion compensation unit 44 may generate a motion compensated

predictive block from one or more reference blocks of the frames stored in the DPB 64. The motion compensation unit 44 may also apply one or more interpolation filters to the predictive block to calculate sub-integer pixel values for use in motion estimation.

[0062] The summer 62 adds the reconstructed residual block to the motion compensated predictive block produced by the motion compensation unit 44 to produce a reference block for storage in the DPB 64. The reference block may then be used by the intra BC unit 48, the motion estimation unit 42 and the motion compensation unit 44 as a predictive block to inter predict another video block in a subsequent video frame.

[0063] FIG. 3 is a block diagram illustrating an exemplary video decoder 30 in accordance with some implementations of the present application. The video decoder 30 includes a video data memory 79, an entropy decoding unit 80, a prediction processing unit 81, an inverse quantization unit 86, an inverse transform processing unit 88, a summer 90, and a DPB 92. The prediction processing unit 81 further includes a motion compensation unit 82, an intra prediction unit 84, and an intra BC unit 85. The video decoder 30 may perform a decoding process generally reciprocal to the encoding process described above with respect to the video encoder 20 in connection with FIG. 2. For example, the motion compensation unit 82 may generate prediction data based on motion vectors received from the entropy decoding unit 80, while the intra-prediction unit 84 may generate prediction data based on intra-prediction mode indicators received from the entropy decoding unit 80.

[0064] In some examples, a unit of the video decoder 30 may be tasked to perform the implementations of the present application. Also, in some examples, the implementations of the present disclosure may be divided among one or more of the units of the video decoder 30. For example, the intra BC unit 85 may perform the implementations of the present application, alone, or in combination with other units of the video decoder 30, such as the motion compensation unit 82, the intra prediction unit 84, and the entropy decoding unit 80. In some examples, the video decoder 30 may not include the intra BC unit 85 and the functionality of intra BC unit 85 may be performed by other components of the prediction processing unit 81, such as the motion compensation unit 82.

[0065] The video data memory 79 may store video data, such as an encoded video bitstream, to be decoded by the other components of the video decoder 30. The video data stored in the video data memory 79 may be obtained, for example, from the storage device 32, 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 (e.g., a flash drive or hard disk). The video data memory 79 may include a Coded Picture Buffer (CPB) that stores encoded video data from an

encoded video bitstream. The DPB 92 of the video decoder 30 stores reference video data for use in decoding video data by the video decoder 30 (e.g., in intra or inter predictive coding modes). The video data memory 79 and the DPB 92 may be formed by any of a variety of memory devices, such as dynamic random access memory (DRAM), including Synchronous DRAM (SDRAM), Magneto-resistive RAM (MRAM), Resistive RAM (RRAM), or other types of memory devices. For illustrative purpose, the video data memory 79 and the DPB 92 are depicted as two distinct components of the video decoder 30 in FIG. 3. But it will be apparent to one skilled in the art that the video data memory 79 and the DPB 92 may be provided by the same memory device or separate memory devices. In some examples, the video data memory 79 may be on-chip with other components of the video decoder 30, or off-chip relative to those components.

[0066] During the decoding process, the video decoder 30 receives an encoded video bitstream that represents video blocks of an encoded video frame and associated syntax elements. The video decoder 30 may receive the syntax elements at the video frame level and/or the video block level. The entropy decoding unit 80 of the video decoder 30 entropy decodes the bitstream to generate quantized coefficients, motion vectors or intra-prediction mode indicators, and other syntax elements. The entropy decoding unit 80 then forwards the motion vectors or intra-prediction mode indicators and other syntax elements to the prediction processing unit 81.

[0067] When the video frame is coded as an intra predictive coded (I) frame or for intra coded predictive blocks in other types of frames, the intra prediction unit 84 of the prediction processing unit 81 may generate prediction data for a video block of the current video frame based on a signaled intra prediction mode and reference data from previously decoded blocks of the current frame.

[0068] When the video frame is coded as an inter-predictive coded (i.e., B or P) frame, the motion compensation unit 82 of the prediction processing unit 81 produces one or more predictive blocks for a video block of the current video frame based on the motion vectors and other syntax elements received from the entropy decoding unit 80. Each of the predictive blocks may be produced from a reference frame within one of the reference frame lists. The video decoder 30 may construct the reference frame lists, List 0 and List 1, using default construction techniques based on reference frames stored in the DPB 92.

[0069] In some examples, when the video block is coded according to the intra BC mode described herein, the intra BC unit 85 of the prediction processing unit 81 produces predictive blocks for the current video block based on block vectors and other syntax elements received

from the entropy decoding unit 80. The predictive blocks may be within a reconstructed region of the same picture as the current video block defined by the video encoder 20.

[0070] The motion compensation unit 82 and/or the intra BC unit 85 determines prediction information for a video block of the current video frame by parsing the motion vectors and other syntax elements, and then uses the prediction information to produce the predictive blocks for the current video block being decoded. For example, the motion compensation unit 82 uses some of the received syntax elements to determine a prediction mode (e.g., intra or inter prediction) used to code video blocks of the video frame, an inter prediction frame type (e.g., B or P), construction information for one or more of the reference frame lists for the frame, motion vectors for each inter predictive encoded video block of the frame, inter prediction status for each inter predictive coded video block of the frame, and other information to decode the video blocks in the current video frame.

[0071] Similarly, the intra BC unit 85 may use some of the received syntax elements, e.g., a flag, to determine that the current video block was predicted using the intra BC mode, construction information of which video blocks of the frame are within the reconstructed region and should be stored in the DPB 92, block vectors for each intra BC predicted video block of the frame, intra BC prediction status for each intra BC predicted video block of the frame, and other information to decode the video blocks in the current video frame.

[0072] The motion compensation unit 82 may also perform interpolation using the interpolation filters as used by the video encoder 20 during encoding of the video blocks to calculate interpolated values for sub-integer pixels of reference blocks. In this case, the motion compensation unit 82 may determine the interpolation filters used by the video encoder 20 from the received syntax elements and use the interpolation filters to produce predictive blocks.

[0073] The inverse quantization unit 86 inverse quantizes the quantized transform coefficients provided in the bitstream and entropy decoded by the entropy decoding unit 80 using the same quantization parameter calculated by the video encoder 20 for each video block in the video frame to determine a degree of quantization. The inverse transform processing unit 88 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 reconstruct the residual blocks in the pixel domain.

[0074] After the motion compensation unit 82 or the intra BC unit 85 generates the predictive block for the current video block based on the vectors and other syntax elements, the summer 90 reconstructs decoded video block for the current video block by summing the residual block from the inverse transform processing unit 88 and a corresponding predictive

block generated by the motion compensation unit 82 and the intra BC unit 85. An in-loop filter 91 such as deblocking filter, SAO filter, CCSAO filter and/or ALF may be positioned between the summer 90 and the DPB 92 to further process the decoded video block. In some examples, the in-loop filter 91 may be omitted, and the decoded video block may be directly provided by the summer 90 to the DPB 92. The decoded video blocks in a given frame are then stored in the DPB 92, which stores reference frames used for subsequent motion compensation of next video blocks. The DPB 92, or a memory device separate from the DPB 92, may also store decoded video for later presentation on a display device, such as the display device 34 of FIG. 1.

[0075] In a typical video coding process, a video sequence typically includes an ordered set of frames or pictures. Each frame may include three sample arrays, denoted SL, SCb, and SCr. SL is a two-dimensional array of luma samples. SCb is a two-dimensional array of Cb chroma samples. SCr is a two-dimensional array of Cr chroma samples. In other instances, a frame may be monochrome and therefore includes only one two-dimensional array of luma samples.

[0076] As shown in FIG. 4A, the video encoder 20 (or more specifically the partition unit 45) generates an encoded representation of a frame by first partitioning the frame into a set of CTUs. A video frame may include an integer number of CTUs ordered consecutively in a raster scan order from left to right and from top to bottom. Each CTU is a largest logical coding unit and the width and height of the CTU are signaled by the video encoder 20 in a sequence parameter set, such that all the CTUs in a video sequence have the same size being one of 128×128, 64×64, 32×32, and 16×16. But it should be noted that the present application is not necessarily limited to a particular size. As shown in FIG. 4B, each CTU may comprise one CTB of luma samples, two corresponding coding tree blocks of chroma samples, and syntax elements used to code the samples of the coding tree blocks. The syntax elements describe properties of different types of units of a coded block of pixels and how the video sequence can be reconstructed at the video decoder 30, including inter or intra prediction, intra prediction mode, motion vectors, and other parameters. In monochrome pictures or pictures having three separate color planes, a CTU may comprise a single coding tree block and syntax elements used to code the samples of the coding tree block. A coding tree block may be an NxN block of samples.

[0077] To achieve a better performance, the video encoder 20 may recursively perform tree partitioning such as binary-tree partitioning, ternary-tree partitioning, quad-tree partitioning or a combination thereof on the coding tree blocks of the CTU and divide the CTU into smaller

CUs. As depicted in FIG. 4C, the 64x64 CTU 400 is first divided into four smaller CUs, each having a block size of 32x32. Among the four smaller CUs, CU 410 and CU 420 are each divided into four CUs of 16x16 by block size. The two 16x16 CUs 430 and 440 are each further divided into four CUs of 8x8 by block size. FIG. 4D depicts a quad-tree data structure illustrating the end result of the partition process of the CTU 400 as depicted in FIG. 4C, each leaf node of the quad-tree corresponding to one CU of a respective size ranging from 32x32 to 8x8. Like the CTU depicted in FIG. 4B, each CU may comprise a CB of luma samples and two corresponding coding blocks of chroma samples of a frame of the same size, and syntax elements used to code the samples of the coding blocks. In monochrome pictures or pictures having three separate color planes, a CU may comprise a single coding block and syntax structures used to code the samples of the coding block. It should be noted that the quad-tree partitioning depicted in FIGS. 4C and 4D is only for illustrative purposes and one CTU can be split into CUs to adapt to varying local characteristics based on quad/ternary/binary-tree partitions. In the multi-type tree structure, one CTU is partitioned by a quad-tree structure and each quad-tree leaf CU can be further partitioned by a binary and ternary tree structure. As shown in FIG. 4E, there are five possible partitioning types of a coding block having a width W and a height H, i.e., quaternary partitioning, horizontal binary partitioning, vertical binary partitioning, horizontal ternary partitioning, and vertical ternary partitioning.

[0078] In some implementations, the video encoder 20 may further partition a coding block of a CU into one or more MxN PBs. A PB is a rectangular (square or non-square) block of samples on which the same prediction, inter or intra, is applied. A PU of a CU may comprise a PB of luma samples, two corresponding PBs of chroma samples, and syntax elements used to predict the PBs. In monochrome pictures or pictures having three separate color planes, a PU may comprise a single PB and syntax structures used to predict the PB. The video encoder 20 may generate predictive luma, Cb, and Cr blocks for luma, Cb, and Cr PBs of each PU of the CU.

[0079] The video encoder 20 may use intra prediction or inter prediction to generate the predictive blocks for a PU. If the video encoder 20 uses intra prediction to generate the predictive blocks of a PU, the video encoder 20 may generate the predictive blocks of the PU based on decoded samples of the frame associated with the PU. If the video encoder 20 uses inter prediction to generate the predictive blocks of a PU, the video encoder 20 may generate the predictive blocks of the PU based on decoded samples of one or more frames other than the frame associated with the PU.

[0080] After the video encoder 20 generates predictive luma, Cb, and Cr blocks for one or

more PUs of a CU, the video encoder 20 may generate a luma residual block for the CU by subtracting the CU's predictive luma blocks from its original luma coding block such that each sample in the CU's luma residual block indicates a difference between a luma sample in one of the CU's predictive luma blocks and a corresponding sample in the CU's original luma coding block. Similarly, the video encoder 20 may generate a Cb residual block and a Cr residual block for the CU, respectively, such that each sample in the CU's Cb residual block indicates a difference between a Cb sample in one of the CU's predictive Cb blocks and a corresponding sample in the CU's original Cb coding block and each sample in the CU's Cr residual block may indicate a difference between a Cr sample in one of the CU's predictive Cr blocks and a corresponding sample in the CU's original Cr coding block.

[0081] Furthermore, as illustrated in FIG. 4C, the video encoder 20 may use quad-tree partitioning to decompose the luma, Cb, and Cr residual blocks of a CU into one or more luma, Cb, and Cr transform blocks respectively. A transform block is a rectangular (square or non-square) block of samples on which the same transform is applied. A TU of a CU may comprise a transform block of luma samples, two corresponding transform blocks of chroma samples, and syntax elements used to transform the transform block samples. Thus, each TU of a CU may be associated with a luma transform block, a Cb transform block, and a Cr transform block. In some examples, the luma transform block associated with the TU may be a sub-block of the CU's luma residual block. The Cb transform block may be a sub-block of the CU's Cb residual block. The Cr transform block may be a sub-block of the CU's Cr residual block. In monochrome pictures or pictures having three separate color planes, a TU may comprise a single transform block and syntax structures used to transform the samples of the transform block.

[0082] The video encoder 20 may apply one or more transforms to a luma transform block of a TU to generate a luma coefficient block for the TU. A coefficient block may be a two-dimensional array of transform coefficients. A transform coefficient may be a scalar quantity. The video encoder 20 may apply one or more transforms to a Cb transform block of a TU to generate a Cb coefficient block for the TU. The video encoder 20 may apply one or more transforms to a Cr transform block of a TU to generate a Cr coefficient block for the TU.

[0083] After generating a coefficient block (e.g., a luma coefficient block, a Cb coefficient block or a Cr coefficient block), the video encoder 20 may quantize the coefficient block. Quantization generally refers to a process in which transform coefficients are quantized to possibly reduce the amount of data used to represent the transform coefficients, providing further compression. After the video encoder 20 quantizes a coefficient block, the video

encoder 20 may entropy encode syntax elements indicating the quantized transform coefficients. For example, the video encoder 20 may perform CABAC on the syntax elements indicating the quantized transform coefficients. Finally, the video encoder 20 may output a bitstream that includes a sequence of bits that forms a representation of coded frames and associated data, which is either saved in the storage device 32 or transmitted to the destination device 14.

[0084] After receiving a bitstream generated by the video encoder 20, the video decoder 30 may parse the bitstream to obtain syntax elements from the bitstream. The video decoder 30 may reconstruct the frames of the video data based at least in part on the syntax elements obtained from the bitstream. The process of reconstructing the video data is generally reciprocal to the encoding process performed by the video encoder 20. For example, the video decoder 30 may perform inverse transforms on the coefficient blocks associated with TUs of a current CU to reconstruct residual blocks associated with the TUs of the current CU. The video decoder 30 also reconstructs the coding blocks of the current CU by adding the samples of the predictive blocks for PUs of the current CU to corresponding samples of the transform blocks of the TUs of the current CU. After reconstructing the coding blocks for each CU of a frame, video decoder 30 may reconstruct the frame.

[0085] As noted above, video coding achieves video compression using primarily two modes, i.e., intra-frame prediction (or intra-prediction) and inter-frame prediction (or inter-prediction). It is noted that IBC could be regarded as either intra-frame prediction or a third mode. Between the two modes, inter-frame prediction contributes more to the coding efficiency than intra-frame prediction because of the use of motion vectors for predicting a current video block from a reference video block.

[0086] But with the ever-improving video data capturing technology and more refined video block size for preserving details in the video data, the amount of data required for representing motion vectors for a current frame also increases substantially. One way of overcoming this challenge is to benefit from the fact that not only a group of neighboring CUs in both the spatial and temporal domains have similar video data for predicting purpose but the motion vectors between these neighboring CUs are also similar. Therefore, it is possible to use the motion information of spatially neighboring CUs and/or temporally co-located CUs as an approximation of the motion information (e.g., motion vector) of a current CU by exploring their spatial and temporal correlation, which is also referred to as "Motion Vector Predictor (MVP)" of the current CU.

[0087] Instead of encoding, into the video bitstream, an actual motion vector of the current

CU determined by the motion estimation unit 42 as described above in connection with FIG. 2, the motion vector predictor of the current CU is subtracted from the actual motion vector of the current CU to produce a Motion Vector Difference (MVD) for the current CU. By doing so, there is no need to encode the motion vector determined by the motion estimation unit 42 for each CU of a frame into the video bitstream and the amount of data used for representing motion information in the video bitstream can be significantly decreased.

[0088] Like the process of choosing a predictive block in a reference frame during inter-frame prediction of a code block, a set of rules need to be adopted by both the video encoder 20 and the video decoder 30 for constructing a motion vector candidate list (also known as a "merge list") for a current CU using those potential candidate motion vectors associated with spatially neighboring CUs and/or temporally co-located CUs of the current CU and then selecting one member from the motion vector candidate list as a motion vector predictor for the current CU. By doing so, there is no need to transmit the motion vector candidate list itself from the video encoder 20 to the video decoder 30 and an index of the selected motion vector predictor within the motion vector candidate list is sufficient for the video encoder 20 and the video decoder 30 to use the same motion vector predictor within the motion vector candidate list for encoding and decoding the current CU.

[0089] In the Enhanced Compression Model (ECM), cross component prediction techniques are utilized to improve the compression efficiency of chroma components. Example cross component prediction techniques and their improved variants are introduced herein.

[0090] A brief description of CCLM prediction is provided below. To reduce the cross component redundancy, a CCLM prediction mode is used in the VVC, where chroma samples of a video block (e.g., a CU) are predicted based on reconstructed luma samples (e.g., $\mathrm{rec}_L(i,j)$) of the CU by using a filtering function of a linear model as follows:

$$\mathrm{pred}_C(i,j) = \alpha \cdot \mathrm{rec}_L'(i,j) + \beta. \qquad (1)$$

[0091] In the above expression (1), $\mathrm{pred}_C(i,j)$ represents a prediction chroma sample in a sample position (i,j) of the CU; $\mathrm{rec}_L'(i,j)$ represents a downsampled reconstructed luma sample of the CU which is obtained by performing down-sampling on a reconstructed luma sample $\mathrm{rec}_L(i,j)$ in the sample position (i,j) of the CU; and α and β are linear model coefficients (also referred to as linear model parameters) which are derived from at most four neighbouring chroma samples and their corresponding downsampled luma samples. Suppose that a current chroma block has a size of W×H, where W represents a width and H represents a height of the chroma block. Then, an adjusted width W' and an adjusted H' can be obtained

as follows using expressions (2)-(4):

$$W' = W, H' = H, \text{ when a linear model (LM) mode is applied;} \quad (2)$$

$$W' = W + H, \text{ when an LM_A mode is applied;} \quad (3)$$

$$H' = H + W, \text{ when an LM_L mode is applied.} \quad (4)$$

[0092] In the LM mode, above samples and left samples of the CU are used together to calculate the linear model parameters α and β. The above samples can be samples on top of (or above) the CU. The left samples can be samples on the left of the CU. LM_A may denote a linear model with above samples. In the LM_A mode, only the above samples of the CU are used to calculate the linear model parameters α and β. LM_L may denote a linear model with left samples. In the LM_L mode, only the left samples of the CU are used to calculate the linear model parameters α and β.

[0093] Based on the adjusted width W' and the adjusted height H', locations of above samples of a chroma block are denoted as S[0, −1], …, S[W' − 1, −1], and locations of left samples of the chroma block are denoted as S[−1, 0], …, S[−1, H' − 1]. Positions of four neighbouring chroma samples are selected according to three cases. In a first case, S[W' / 4, −1], S[3 * W' / 4, −1], S[−1, H' / 4], and S[−1, 3 * H' / 4] are selected as the positions of the four neighbouring chroma samples when (i) the LM mode is applied and (ii) both the above samples and the left samples are available. In a second case, S[W' / 8, −1], S[3 * W' / 8, −1], S[5 * W' / 8, −1], and S[7 * W' / 8, −1] are selected as the positions of the four neighbouring chroma samples when (i) the LM_A mode is applied and (ii) the above samples are available (or when only the above samples are available). In a third case, S[−1, H' / 8], S[−1, 3 * H' / 8], S[−1, 5 * H' / 8], and S[−1, 7 * H' / 8] are selected as the positions of the four neighbouring chroma samples when (i) the LM_L mode is applied and (ii) the left samples are available (or when only the left samples are available).

[0094] Four neighbouring luma samples corresponding to the selected locations (e.g., the selected locations in the above first, second or third case) are obtained by a down-sampling operation. For example, neighboring reconstructed luma samples are downsampled to obtain four neighbouring luma samples corresponding to the selected locations, so that the obtained four neighbouring luma samples can be downsampled reconstructed luma samples corresponding to the selected locations. Then, the obtained four neighboring luma samples are compared four times among themselves to find two larger values (denoted as x^0_A and x^1_A) and two smaller values (denoted as x^0_B and x^1_B) among themselves. Chroma sample values corresponding to the two larger values and the two smaller values are denoted as y^0_A, y^1_A, y^0_B,

and y^1_B, respectively. Then, X_a, X_b, Y_a and Y_b are derived as follows using expressions (5)-(8):

$$X_a=(x^0_A + x^1_A +1)>>1; \quad (5)$$

$$X_b=(x^0_B + x^1_B +1)>>1; \quad (6)$$

$$Y_a=(y^0_A + y^1_A +1)>>1; \quad (7)$$

$$Y_b=(y^0_B + y^1_B +1)>>1. \quad (8)$$

[0095] In the above expressions (5)-(8), ">>" represents a right shift operator. Subsequently, the linear model parameters α and β are obtained according to the following expressions (9) and (10):

$$\alpha = \frac{Y_a - Y_b}{X_a - X_b}, \quad (9)$$

$$\beta = Y_b - \alpha \cdot X_b. \quad (10)$$

[0096] FIG. 5 shows a diagram of locations of left samples and above samples of a CU involved in the CCLM mode. For example, FIG. 5 shows locations of left samples 502 and locations of above samples 504 of an N×N chroma block 501 in the CU. FIG. 5 also shows locations of downsampled left samples 506 and downsampled above samples 508 of an 2N× 2N luma block 503 in the CU.

[0097] The parameter computation described above can be performed as part of the decoding process. Therefore, there is no need to signal a syntax element to convey values of α and β from video encoder 20 to video decoder 30.

[0098] The CCLM included in the VVC can be extended by adding three Multi-Model Linear Model (MMLM) modes (JVET-D0110). In each MMLM mode, the neighboring reconstructed samples are classified into two classes using a threshold which is an average of the neighboring reconstructed luma samples. The parameters of a linear model in each class can be derived using the Least-Mean-Square (LMS) method. For the CCLM mode, the LMS method is also used to derive the parameters of the linear model. A slope adjustment can be applied to the CCLM prediction and the MMLM prediction. The adjustment is configured to tilt a linear function which maps luma values to chroma values with respect to a center point determined by an average luma value of the reference samples.

[0099] With respect to the slope adjustment of the CCLM, the CCLM may use a model with 2 parameters to map luma values to chroma values. A slope parameter "a" and a bias parameter "b" can be used to determine a filtering function as follows (as shown in the left

graph of FIG. 6):

$$chromaVal = a * lumaVal + b. \quad (11)$$

[00100] In the above expression (11), chromaVal and lumaVal may represent a chroma sample value and a luma sample value, respectively. An adjustment "u" to the slope parameter "a" is signaled to update the filtering function of the model to a form as follows (as shown with a solid line in the right graph of FIG. 6):

$$chromaVal = a' * lumaVal + b'. \quad (12)$$

[00101] In the above expression (12), $a' = a + u$, and $b' = b - u * y_r$. y_r represents an average luma value of the reference samples. With this adjustment, the filtering function is tilted or rotated around the point having the average luma value yr. The average luma value y_r of the reference samples can provide a meaningful modification to the model, as shown in FIG. 6.

[00102] The slope adjustment parameter "u" can be provided as an integer between -4 and 4, inclusive, and signaled in the bitstream. The unit of the slope adjustment parameter is 1/8th of a chroma sample value per one luma sample value (for 10-bit content). The slope adjustment is available for the CCLM models that use both above reference samples and left reference samples of the video block ("LM_CHROMA_IDX" and "MMLM_CHROMA_IDX"), but not for the "single side" modes (e.g., when only above reference samples or only left reference samples are available). This adjustment is based on the trade-off consideration between coding efficiency and complexity. When the slope adjustment is applied for a multi-model CCLM mode, both models in the two classes of the multi-model CCLM mode can be adjusted, and thus up to two slope updates (e.g., up to two slope adjustment parameters) are signaled for a single chroma block.

[00103] On the encoder side, a Sum of Absolute Transformed Difference (SATD) based search can be performed to search for the best value of the slope update for Cr, and a similar SATD based search can be performed to search for the best value of the slope update for Cb. If either search leads to a non-zero slope adjustment parameter, a combined slope adjustment pair (e.g., including the SATD based update for Cr and the SATD based update for Cb) is included in the list of Rate Distortion (RD) checks for the TU.

[00104] A convolutional cross component intra prediction model is provided herein. In this case, a convolutional cross component model (CCCM) is applied to predict chroma samples from reconstructed luma samples like that in the CCLM modes. Like the CCLM, the reconstructed luma samples are downsampled to match a lower resolution chroma grid when chroma sub-sampling is used. Similar to the CCLM, top reference samples, left reference samples, or both top and left reference samples, are used as templates for model derivation.

[00105] Also similar to the CCLM, there is an option of using a single model or multi-model variant of CCCM. The multi-model variant uses two models, including a first model derived for samples being greater than an average luma reference value (e.g., an average luma value of the reference samples) and a second model for the rest of the samples, like that of the CCLM design. The multi-model CCCM mode can be selected for PUs which have at least 128 reference samples available.

[00106] A convolutional filter (e.g., a convolutional 7-tap filter) can be applied in the CCCM mode, which includes (i) a 5-tap plus sign shape spatial component, (ii) a non-linear term P_{NL}, and (iii) a bias term. As shown in FIG. 7, the input to the 5-tap spatial component of the filter may include: (i) a center (C) luma sample which is collocated with a chroma sample to be predicted; (ii) a north (N) luma sample above the center luma sample; (iii) a south (S) luma sample below the center luma sample; (iv) a west (W) luma sample on the left of the center luma sample; and (v) an east (E) luma sample on the right of the center luma sample. The north, south, west and east luma samples are neighboring samples of the center luma sample.

[00107] The non-linear term P_{NL} is represented as power of two of the center luma sample C and scaled to the sample value range of the content as follow:

$$P_{NL} = P(C) = (C*C + midVal) >> bitDepth. \quad (13)$$

[00108] In the above expression (13), midVal represents a middle chroma value, and bitDepth represents the sample value range of the content. For example, for 10-bit content, midVal = 512, bitDepth = 10, and the above expression (13) can be rewritten as follows in expression (14):

$$P_{NL} = P(C) = (C*C + 512) >> 10. \quad (14)$$

[00109] The bias term B represents a scalar offset between the input and the output (similar to the offset term in CCLM) and can be set to the middle chroma value (e.g., 512 for 10-bit content). As a result, the output of the filter applied in the CCCM mode can be calculated as a convolution between the filter coefficients c_i (e.g., c_0, c_1, c_2, c_3, c_4, c_5, c_6) and the input values (e.g., C, N, S, E, W, P_{NL}, B), and clipped to the range of valid chroma samples. For example, a filtering function in the CCCM mode can be written as follows:

$$predChromaVal = c_0C + c_1N + c_2S + c_3E + c_4W + c_5P + c_6B. \quad (15)$$

[00110] In the above expression (15), predChromaVal represents a value of a prediction chroma sample (e.g., a prediction value of a chroma sample); C represents a center luma sample which is collocated with the chroma sample to be predicted; N, S, E, and W represent the north, south, west and east luma samples of the center luma sample, respectively; P_{NL} represents the

non-linear term; and B represents the bias term.

[00111] Calculation of the filter coefficients c_i (e.g., c_0, c_1, c_2, c_3, c_4, c_5, c_6) are described below. Specifically, the filter coefficients can be calculated by minimizing a mean squared error (MSE) between prediction chroma samples and reconstructed chroma samples in a reference area. For example, as shown in FIG. 8, a current video block 801 may have a current template 802. A reference block 803 corresponding to video block 801 is shown in FIG. 8, and has a reference template 804 corresponding to current template 802. The template 802 and reference template 804 are utilized to derive the filter coefficients. The filter coefficients can be calculated by minimizing an MSE between (i) prediction chroma samples calculated based on the expression (15) for the chroma samples in reference template 804 and (ii) the reconstructed chroma samples in current template 802.

[00112] In another example, FIG. 9 illustrates a reference area 901 which includes 6 lines of chroma samples above a PU 900 and 6 lines of chroma samples on the left of PU 900. Reference area 901 may extend by a width of one line of chroma samples to the right of the PU boundary and by a height of one line of chroma samples below the PU boundary. Extensions to reference area 901 are illustrated by shaded areas 904. Reference area 901 may be adjusted to include only available samples. Extensions 904 to reference area 901 are needed to support the "side samples" of the plus sign shape spatial filter and are padded when samples in extensions 904 are unavailable.

[00113] The MSE minimization can be performed by calculating an autocorrelation matrix for the luma input and a cross-correlation vector between the luma input and the chroma output. The autocorrelation matrix is LDL decomposed (where L represents a lower unit triangular matrix, and D represents a diagonal matrix), and the filter coefficients are calculated using back-substitution. The process is similar to the calculation of the ALF filter coefficients in the ECM, where the LDL decomposition rather than the Cholesky decomposition is chosen to avoid using square root operations.

[00114] The autocorrelation matrix can be calculated using the reconstructed values of luma and chroma samples. These samples are full range (e.g., between 0 and 1023 for 10-bit content), resulting in relatively large values in the autocorrelation matrix. This requires a high bit depth operation during the model parameter calculation. However, fixed offsets can be removed from luma and chroma samples in each PU for each model. This can reduce the magnitudes of the values used in the model parameter calculation, and allows reducing the precision needed for the fixed-point arithmetic. As a result, the 16-bit decimal precision is provided instead of the 22-bit precision of the original CCCM implementation.

[00115] Reference sample values which are outside the top-left corner of the PU are used as the offsets (offsetLuma, offsetCb, and offsetCr) for simplicity. The sample values used in both model parameter calculation and final prediction (i.e., including luma and chroma samples in the reference area, and luma samples in the current PU) are reduced by these offset values according to following expressions (16)-(22):

$$C' = C - \text{offsetLuma}, \quad (16)$$

$$N' = N - \text{offsetLuma}, \quad (17)$$

$$S' = S - \text{offsetLuma}, \quad (18)$$

$$E' = E - \text{offsetLuma}, \quad (19)$$

$$W' = W - \text{offsetLuma}, \quad (20)$$

$$P_{NL}' = \text{nonLinear}(C') = (C'* C' + \text{midVal}) >> \text{bitDepth}, \quad (21)$$

$$B = \text{midVal} = 1 << (\text{bitDepth} - 1). \quad (22)$$

[00116] Then, the chroma value is predicted using a filtering function determined by the following expression (23):

$$\text{predChromaVal} = c_0C' + c_1N' + c_2S' + c_3E' + c_4W' + c_5P_{NL}' + c_6B + \text{offsetChroma}. \quad (23)$$

[00117] In the above expression (23), predChromaVal represents a value of the prediction chroma sample. For the Cr component, offsetChroma is equal to offsetCr. For the Cb component, offsetChroma is equal to offsetCb. To avoid any additional sample level operations, the luma offset is removed during the luma reference sample interpolation. This can be done, for example, by substituting the rounding term used in the luma reference sample interpolation with an updated offset including both the rounding term and the offsetLuma. The chroma offset can be removed by deducting the chroma offset directly from the reference chroma samples. Alternatively, impact of the chroma offset can be removed from the cross component vector giving the identical result. In order to add the chroma offset back to the output of the convolutional prediction operation, the chroma offset is added to the bias term of the convolutional model.

[00118] The process of the CCCM model parameter calculation needs division operations. However, division operations are not always considered implementation friendly. Consistent with some implementations, the division operations may be replaced with the multiplication (with a scale factor) and shift operation, where the scale factor and the number of shifts are calculated based on denominator similar to the method used in the calculation of CCLM parameters.

[00119] A brief description of a Gradient Linear Model (GLM) is also provided herein. For the YUV 4:2:0 color format, a GLM method can be used to predict the chroma samples from

luma sample gradients. Two modes are supported in the GLM method, including a two-parameter GLM mode and a three-parameter GLM mode. Compared with the CCLM mode, the two-parameter GLM mode utilizes luma sample gradients to derive a function of a linear model instead of down-sampling the luma values. Specifically, the input to the CCLM mode (i.e., the downsampled luma samples L) are replaced by luma sample gradients G when the two-parameter GLM mode is applied. The other parts of the CCLM mode (e.g., parameter derivation, prediction sample linear transform) are kept unchanged in the two-parameter GLM mode. For example, a filtering function of the two-parameter GLM mode can be determined by the following expression (24):

$$C_Pred = \alpha \cdot G + \beta. \qquad (24)$$

[00120] In the above expression (24), C_Pred represents a prediction value of a chroma sample (e.g., a value of a prediction chroma sample), G represents a luma sample gradient, and α and β represent model parameters.

[00121] In the three-parameter GLM mode, a chroma sample can be predicted based on both the luma sample gradients and downsampled luma values with different model parameters. The model parameters of the three-parameter GLM mode are derived from 6 rows and 6 columns of adjacent samples by the MSE minimization method based on the LDL decomposition as used in the CCCM mode. For example, a filtering function of the three-parameter GLM mode can be determined by the following expression (25):

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \alpha_2 \cdot \beta. \qquad (25)$$

[00122] In the above expression (25), α_0, α_1, α_2, and β represents model parameters. G represents a luma sample gradient. L represents a downsampled luma value.

[00123] For signaling, when the CCLM mode is enabled to the current CU, a flag is signaled to indicate whether GLM is enabled for both the Cb and Cr components. If the GLM is enabled, another flag is signaled to indicate which of the two GLM modes is selected, and a syntax element is further signaled to select one of four gradient filters for the gradient calculation. The four gradient filters for the GLM are illustrated in FIG. 10.

[00124] With respect to the bitstream signaling, a mode usage is signaled with a CABAC coded PU level flag. A new CABAC context can be included to support this signaling. In the signaling, CCCM is considered as a sub-mode of CCLM. That is, the CCCM flag is only signaled if the intra prediction mode is LM_CHROMA.

[00125] A brief description of the CCCM mode using non-downsampled luma samples is provided herein. In some applications, the CCCM mode is applied using non-downsampled

luma samples (e.g., the luma samples are used directly without a down-sampling operation). This CCCM mode can also be referred to as CCCM without down-sampling. For example, the CCCM mode with a 3×2 filter using non-downsampled luma samples can be applied, which includes 6-tap spatial terms, four non-linear terms, and a bias term. As shown in FIG. 11, the 6-tap spatial terms correspond to 6 neighboring luma samples (i.e., L_0, L_1, ..., L_5) around a center luma sample C which is collocated with the chroma sample to be predicted. The four non-linear terms are derived from the luma samples L_0, L_1, L_2, and L_3. A filtering function of this CCCM mode using non-downsampled luma samples can be expressed as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \text{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \text{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \text{offsetChroma}. \quad (26)$$

[00126] In the above expression (26), α_i represent a coefficient (e.g., i = 0, 1, ..., 9), β represents an offset. $\sum_{i=0}^{5} \alpha_i \cdot (L_i - \text{offsetLuma})$ represent the 6-tap spatial terms. $\sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \text{offsetLuma})^2 + \beta) \gg bitDepth)$ represent the four non-linear terms. Like the previous CCCM design with downsampled luma samples, up to 6 rows of chroma samples above the current CU and 6 columns of chroma samples on the left to the current CU are applied to derive the model parameters. The model parameters are derived based on the same LDL decomposition method used in the previous CCCM design. The CCCM mode using non-downsampled luma samples is signaled as an additional CCCM mode. When the CCCM mode is selected, a flag is signaled and used for both two chroma components to indicate whether the default CCCM mode using downsampled luma samples or the CCCM mode using non-downsampled luma samples is applied. Additionally, the sequence parameter set (SPS) signaling is introduced to indicate whether the CCCM mode using non-downsampled luma samples is enabled.

[00127] With respect to the Gradient and Location based convolutional cross component model (GL-CCCM), luma values can be mapped into chroma values using a filter with inputs including a spatial luma sample, two gradient values, two sets of location information, a non-linear term, and a bias term. The GL-CCCM mode uses gradient and location information instead of the 4 spatial neighbor samples used in the filter of the CCCM mode. A filtering function of the GL-CCCM mode used for the chroma prediction can be written as follows:

$$predChromaVal = c_0 C + c_1 G_y + c_2 G_x + c_3 Y + c_4 X + c_5 P_{NL} + c_6 B. \quad (27)$$

[00128] In the above expression (27), C represents a center spatial luma sample; G_y and G_x represent two gradient values, e.g., a vertical gradient and a horizontal gradient, respectively; Y and X represent two sets of location information, e.g., spatial coordinates of the center luma

sample C such as a vertical coordinate and a horizontal coordinate of the center luma sample, respectively; P_{NL} represents a non-linear term; and B represents a bias term. With reference to FIG. 12, G_y and G_x can be calculated as follows:

$$G_y = (2N + NW + NE) - (2S + SW + SE), \quad (28)$$

$$G_x = (2W + NW + SW) - (2E + NE + SE). \quad (29)$$

[00129] In the above expressions (28) and (29), N, NW, NE, S, SW, SE, W, NW, and E represent a north luma sample, a northwest luma sample, a northeast luma sample, a south luma sample, a southwest luma sample, a southeast luma sample, a west luma sample, and an east luma sample neighboring the center luma sample C, respectively, as shown in FIG. 12. The rest of the model parameters in the expression (27) are the same as those in the expression (15) described above. A reference area for the parameter calculation can be the same as the CCCM mode described above.

[00130] The usage of the GL-CCCM mode is signaled with a CABAC coded PU level flag. In the signaling, GL-CCCM is considered as a sub-mode of CCCM. That is, the GL-CCCM flag is only signaled if an original CCCM flag is true.

[00131] Similar to the CCCM mode, the GL-CCCM mode has 6 modes for calculating the parameters, including: (i) a single-model GL-CCCM from above template samples and left template samples; (i) a single-model GL-CCCM from above template samples; (iii) a single-model GL-CCCM from left template samples; (iv) a multi-model GL-CCCM from above template samples and left template samples; (v) a multi-model GL-CCCM from above template samples; and (vi) a multi-model GL-CCCM from left template samples. In some examples, video encoder 20 performs an SATD based search for the 6 GL-CCCM modes along with the CCCM modes to find the best candidates for full RD tests.

[00132] With respect to the CCCM mode with multiple down-sampling filters, the multiple down-sampling filters can be applied to a group of reconstructed luma samples. A linear combination of these downsampled reconstructed samples is multiplied by the derived filter coefficients to form the final chroma predictor. The horizontal and vertical locations of the center luma sample are also considered in the tested model. The cross component models (e.g., Model 1, Model 2, Model 3) shown in the following expressions (30)-(32) are tested as additional CCCM modes with a mode index signaled in the bitstream. For example, filtering functions of these models can be written as follows:

$$\text{Model 1: predChroma} = c_0 * H(C) + c_1 * G1(C) + c_2 * G2(C) + c_3 * G3(C) + c_4 * P(H(C)) + c_5 * P(G1(C)) + c_6 * P(G2(C)) + c_7 * X + c_8 * Y + c_9 * B, \quad (30)$$

Model 2: predChroma = c_0 * H(C) + c_1 * H(W) + c_2 * H(E) + c_3 * G1(C) + c_4 * G1(W) + c_5 * G1(E) + c_6 * P(H(C)) + c_7 * P(H(W)) + c_8 * P(H(E)) + c_9 * X + c_{10} * B, (31)

Model 3: predChroma = c_0 * H(C) + c_1 * H(NE) + c_2 * H(SW) + c_3 * G3(C) + c_4 * G3(NE) + c_5 * G3(SW) + c_6 * P(H(C)) + c_7 * P(H(NE)) + c_8 * P(H(SW)) + c_9 * Y + c_{10} * B. (32)

[00133] In the above expressions (30)-(32), H(·), G1(·), G2(·), and G3(·) are various down-sampling filters as shown in FIG. 13. C denotes a center luma sample collocated with the chroma sample to be predicted, and N, S, W, E, NE, SW are north, south, west, east, northeast and southwest luma samples around C, respectively, as shown in FIG. 12; c_i are filter coefficients (e.g., i = 0, 1, ..., 10); P(·) represents a non-linear term and can be calculated using the expression (13); B represents a bias term; and X and Y are the horizontal and vertical locations of the center luma sample with respect to the top-left coordinates of the video block, respectively.

[00134] With respect to local-boosting cross component prediction, prediction samples of MM-CCLM/MM-CCCM can be filtered with neighboring samples. As shown in FIG. 14, a 3×3 low-pass filter is applied to filter prediction samples generated by MM-CCLM/MM-CCCM. For a sample on a top or left boundary, a filtering window may involve neighboring reconstructed samples. For inner samples, the filtering window only involves prediction samples, which may be padded. A flag is signaled to indicate whether filtering is applied for a block coded with MM-CCLM/MM-CCCM.

[00135] A brief description of a Cross Component Prediction (CCP) merge mode is provided herein. Specifically, for chroma coding, a flag is signaled to indicate whether a CCP mode (including CCLM, CCCM, GLM, and their variants) or a non-CCP mode (e.g., a conventional chroma intra prediction mode, a fusion of chroma intra prediction mode) is used. If the CCP mode is selected, another flag is signaled to indicate how to derive the CCP type and parameters, i.e., either from a CCP merge list or signaled/derived on-the-fly. The CCP merge list is constructed from spatial adjacent candidates, spatial non-adjacent candidates, or history-based candidates. After including these candidates in the CCP merge list, default candidates (e.g., default models) may be further included into the merge list if the merge is not full yet. In order to remove redundant CCP models from the merge list, a pruning operation is applied. After constructing the merge list, the CCP models in the merge list are reordered depending on their respective SAD costs which are obtained using a neighboring template of the current block.

[00136] With respect to spatial adjacent candidates and non-adjacent candidates in the CCP merge mode, positions and an inclusion order of the spatial adjacent candidates and the non-adjacent candidates are the same as those determined in the ECM for regular inter merge

prediction candidates. With respect to history-based candidates, a history-based table is maintained to include the recently used CCP models as the history-based candidates, and the table is reset at the beginning of each CTU row. If the current merge list is not full after including the spatial adjacent candidates and non-adjacent candidates, the CCP models in the history-based table are added into the merge list.

[00137] With respect to default candidates, CCLM candidates with default scaling parameters are included into the merge list only when the merge list is not full after including the spatial adjacent candidates, the spatial non-adjacent candidates, or the history-based candidates into the list. If the current merge list has no candidates with the single-model CCLM mode, the default scaling parameters are {0, 1/8, -1/8, 2/8, -2/8, 3/8, -3/8, 4/8, -4/8, 5/8, -5/8, 6/8}. Otherwise, the default scaling parameters are {0, the scaling parameter of the first CCLM candidate + {1/8, -1/8, 2/8, -2/8, 3/8, -3/8, 4/8, -4/8, 5/8, -5/8, 6/8}}. The offset parameter is derived according to the default scaling parameters, an average neighboring reconstructed luma sample value, and an average neighboring reconstructed Cb/Cr sample value.

[00138] A flag is signaled to indicate whether the CCP merge mode is applied or not. If the CCP merge mode is applied, an index is signaled to indicate which candidate model in the merge list is used by the current block. In addition, the CCP merge mode is not allowed for the current chroma coding block when the current CU is coded by intra sub-partitions (ISP) with a single tree, or when a block size of the current chroma coding block is less than or equal to 16.

[00139] The example cross component prediction techniques described above can improve the efficiency of chroma coding in the ECM. However, in these cross component prediction techniques, only reconstructed luma samples are utilized as the input of a cross component prediction model to predict the chroma samples, and other information such as residual luma samples and prediction luma samples are neglected.

[00140] Consistent with some aspects of the present disclosure, the cross component prediction techniques disclosed herein can be improved by incorporating the residual luma samples, the prediction luma samples, or both, into a prediction model to generate prediction chroma samples. Therefore, the coding efficiency can be further improved. For example, as described below in more detail with reference to FIG. 15, the prediction model disclosed herein can be a residual based model (e.g., a residual based CCLM model, a residual based GLM model, or a residual based CCCM model), a prediction based model (e.g., a prediction based CCLM model, a prediction based GLM model, or a prediction based CCCM model), or a prediction and residual based model (e.g., a prediction and residual based CCLM model, a

prediction and residual based GLM model, or a prediction and residual based CCCM model). It is contemplated that the different types of prediction models can be applied individually or combinedly to generate the prediction chroma samples.

[00141] FIG. 15 is a flow chart of an exemplary method 1500 for video coding in accordance with some implementations of the present disclosure. Method 1500 may be implemented by a processor associated with video encoder 20 or video decoder 30 (e.g., method 1500 may be implemented on the encoder side and/or the decoder side), and may include steps 1502-1504 as described below. Some of the steps may be optional to perform the disclosure provided herein. Further, some of the steps may be performed simultaneously, or in a different order than shown in FIG. 15.

[00142] In step 1502, the processor may determine at least one of residual luma sample information or prediction luma sample information associated with a video block from a video frame of a video. For example, the processor may determine residual luma samples associated with the video block. In another example, the processor may determine prediction luma samples associated with the video block. In yet another example, the processor may determine both the residual luma samples and the prediction luma samples associated with the video block. Further, the processor may generate reconstructed luma samples based on the residual luma samples and the prediction luma samples.

[00143] When performed on the encoder side, the processor of encoder 20 may perform operations like those described above with reference to video encoder 20 to generate a predictive luma block including the prediction luma samples for the video block. The processor may also generate a residual luma block by subtracting the predictive luma block from an original luma coding block corresponding to the video block. The residual luma block may include the residual luma samples. The processor may perform transform processing, quantization, and entropy coding on the residual luma samples before sending the residual luma samples to video decoder 30 through a bitstream. Then, the processor may generate a reconstructed residual luma block. The processor may generate a reconstructed luma block based on the reconstructed residual luma block and the predictive luma block. The reconstructed luma block may include reconstructed luma samples for the video block.

[00144] When performed on the decoder side, the processor of video decoder 30 may perform operations like those described above with reference to video decoder 30 to generate a residual luma block based on the bitstream received from video encoder 20. The residual luma block may include residual luma samples for the video block. The processor may also generate a predictive luma block based on syntax elements signaled through the bitstream. The

predictive luma block may include prediction luma samples for the video block. The processor may generate a reconstructed luma block based on the residual luma block and the predictive luma block. The reconstructed luma block may include reconstructed luma samples for the video block.

[00145] In step 1504, the processor may apply a prediction model to generate prediction chroma samples associated with the video block based on at least one of the residual luma sample information or the prediction luma sample information.

[00146] In some implementations, the prediction model can be a residual based model. The processor may apply the prediction model to generate the prediction chroma samples based on the residual luma samples. In some examples, the processor may apply the prediction model to generate the prediction chroma samples based on the residual luma samples and the reconstructed luma samples. The residual luma samples can be downsampled and used in a linear term in a filtering function of the prediction model. Alternatively, the residual luma samples may be downsampled and used in a non-linear term in a filtering function of the prediction model. Alternatively, the residual luma samples can be directly used in a linear term in a filtering function of the prediction model without down-sampling. The residual based model is described below with respect to three example cases.

[00147] In a first example case, the prediction model can be a residual based CCLM model. The residual luma samples may be used in the residual based CCLM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the residual based CCLM model to predict the chroma component.

[00148] For example, the residual luma samples can be applied in a linear term in a filtering function of the residual based CCLM model. Like the reconstructed luma block, the residual luma block is also downsampled. The downsampled residual luma block has the same block size as that of the prediction chroma block. The filtering function of the residual based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{rec_L}'(i,j) + \beta \cdot \mathrm{res_L}'(i,j) + \gamma. \qquad (33)$$

[00149] In the above expression (33), $\mathrm{pred_C}(i,j)$ represents a prediction chroma sample in the video block; $\mathrm{rec_L}'(i,j)$ and $\mathrm{res_L}'(i,j)$ represent a downsampled reconstructed luma sample and a downsampled residual luma sample of the video block which are obtained by performing down-sampling on a reconstructed luma sample $\mathrm{rec_L}(i,j)$ and a residual luma sample $\mathrm{res_L}(i,j)$, respectively; and α, β, and γ represent linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma

samples and downsampled residual luma samples. The downsampled residual luma sample $\mathrm{res_L}'(i,j)$ is applied in the linear term $\beta \cdot \mathrm{res_L}'(i,j)$ in the expression (33).

[00150] In another example, the residual luma samples can be accounted for in a non-linear term in a filtering function of the residual based CCLM model. Like the reconstructed luma block, the residual luma block is also downsampled. The downsampled residual luma block has the same block size as that of the prediction chroma block. The filtering function of the residual based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{rec_L}'(i,j) + \beta \cdot \mathrm{res_L}'(i,j) \cdot \mathrm{res_L}'(i,j) + \gamma. \qquad (34)$$

[00151] In the above expression (34), $\mathrm{pred_C}(i,j)$ represents a prediction chroma sample in the video block; $\mathrm{rec_L}'(i,j)$ and $\mathrm{res_L}'(i,j)$ represent a downsampled reconstructed luma sample and a downsampled residual luma sample of the video block which are obtained by performing down-sampling on a reconstructed luma sample $\mathrm{rec_L}(i,j)$ and a residual luma sample $\mathrm{res_L}(i,j)$, respectively; and α, β, and γ represent linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma samples and downsampled residual luma samples. The downsampled residual luma sample $\mathrm{res_L}'(i,j)$ is used in the non-linear term $\beta \cdot \mathrm{res_L}'(i,j) \cdot \mathrm{res_L}'(i,j)$ in the expression (34).

[00152] In yet another example, the residual luma samples can be directly applied in a linear term in a filtering function of the residual based CCLM model (e.g., without down-sampling the residual luma sample). In this case, the residual luma block is not downsampled, whereas the reconstructed luma block is downsampled. The filtering function of the residual based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{rec_L}'(i,j) + \sum_{k=0}^{5} \alpha_k \cdot R_k\,(i,j) + \gamma. \qquad (35)$$

[00153] In the above expression (35), $\mathrm{pred_C}(i,j)$ represents a prediction chroma sample in the video block, and $\mathrm{rec_L}'(i,j)$ represents a downsampled reconstructed luma sample which is obtained by performing down-sampling on the reconstructed luma samples $\mathrm{rec_L}(i,j)$. $R_k(i,j)$ represents a residual luma sample of the video block as shown in FIG. 16, and α, α_k, and γ are linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma samples and non-downsampled residual luma samples. The residual luma sample $R_k(i,j)$ is applied in the linear term $\sum_{k=0}^{5} \alpha_k \cdot R_k\,(i,j)$ in the expression (35).

[00154] With reference to FIG. 16, "C" represents a center luma sample collocated with the chroma sample to be predicted (e.g., in a sample position (i,j) as shown in the expression (35)). R_k represents a residual luma sample around the center luma sample, which is also referred to

as $R_k(i,j)$ in the expression (35), with k = 0, 1, 2, 3, 4, or 5. R_k is non-downsampled.

[00155] In a second example case, the prediction model can be a residual based GLM model. The residual luma samples may be used in the residual based GLM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the residual based GLM model to predict the chroma component.

[00156] For example, the residual luma samples can be applied in a linear term in a filtering function of the residual based GLM model. Like the reconstructed luma block, the residual luma block is also downsampled. The downsampled residual luma block has the same block size as that of the prediction chroma block. A filtering function of a residual based two-parameter GLM model and a filtering function of a residual based three-parameter GLM model can be written in the following expressions (36) and (37), respectively:

$$C_Pred = \alpha \cdot G + \beta \cdot R + \gamma, \qquad (36)$$

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \alpha_2 \cdot R + \alpha_3 \cdot \beta. \qquad (37)$$

[00157] In the above expressions (36) and (37), C_Pred represents a prediction chrome sample; and G, R, and L represent a luma sample gradient value, a downsampled residual luma sample value, and a downsampled luma sample value, respectively. In some implementations (e.g., on the decoder side), the downsampled luma sample value L may represent a downsampled reconstructed luma sample value.

[00158] In another example, the residual luma samples can be applied in a non-linear term in a filtering function of the residual based GLM model. Like the reconstructed luma block, the residual luma block is also downsampled. The downsampled residual luma block has the same block size as that of the prediction chroma block. A filtering function of a residual based two-parameter GLM model and a filtering function of a residual based three-parameter GLM model can be written in the following expressions (38) and (39), respectively:

$$C_Pred = \alpha \cdot G + \beta \cdot R \cdot R + \gamma, \qquad (38)$$

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \alpha_2 \cdot R \cdot R + \alpha_3 \cdot \beta. \qquad (39)$$

[00159] In the above expressions (38) and (39), C_pred represents a prediction chrome sample; and G, R, and L represent a luma sample gradient value, a downsampled residual luma sample value, and a downsampled luma sample value, respectively.

[00160] In yet another example, the residual luma samples can be directly applied in a linear term in a filtering function of the residual based GLM model (e.g., without down-sampling the residual luma samples). A filtering function of a residual based two-parameter GLM model and a filtering function of a residual based three-parameter GLM model can be written in the

following expressions (40) and (41), respectively:

$$C_Pred = \alpha \cdot G + \sum_{i=0}^{5} \alpha_i \cdot R_i + \gamma, \quad (40)$$

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \sum_{i=0}^{5} \alpha_i \cdot R_i + \alpha_6 \cdot \beta. \quad (41)$$

[00161] In the above expressions (40) and (41), C_Pred represents a prediction chrome sample; and G, R_i, and L represent a luma sample gradient value, a residual luma sample value (without down-sampling), and a downsampled luma sample value, respectively. R_i represents a residual luma sample around the center luma sample C as shown in FIG. 16, with i = 0, 1, 2, 3, 4, or 5.

[00162] In a third example case, the prediction model can be a residual based CCCM model. The residual luma samples may be used in the residual based CCCM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the residual based CCCM model to predict the chroma component.

[00163] For example, the residual luma samples can be applied in a linear term in a filtering function of the residual based CCCM model. Like the reconstructed luma block, the residual luma block is also downsampled. The downsampled residual luma block has the same block size as that of the prediction chroma block. A filtering function of the residual based CCCM model (when the residual luma samples and the luma samples are downsampled) can be written as follows:

$$C_Pred = \alpha_0 \mathrm{C} + \alpha_1 \mathrm{N} + \alpha_2 \mathrm{S} + \alpha_3 \mathrm{E} + \alpha_4 \mathrm{W} + \alpha_5 \mathrm{P}_{NL} + \alpha_6 \mathrm{B} + \alpha_7 \mathrm{R}. \quad (42)$$

[00164] In the above expression (42), C_Pred represents a prediction chroma sample. C, N, S, E, and W represent a center luma sample which is collocated with the chroma sample to be predicted, a north luma sample above the center luma sample, a south luma sample below the center luma sample, an east on the right of the center luma sample, and a west luma sample on the left of the center luma sample, as shown in FIG. 7, and these luma samples are downsampled. α_i represents a filter coefficient with i =0, 1, ..., or 7. $\mathrm{P_{NL}}$ represents a non-linear term. B represents a bias term. R represents a downsampled residual luma sample. In some implementations (e.g., on the decoder side), the downsampled luma samples C, N, S, E, and W may represent corresponding downsampled reconstructed luma samples.

[00165] On the other hand, a filtering function of the residual based CCCM model when the luma samples are not downsampled (but the residual luma samples are downsampled) can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \mathrm{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \mathrm{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \alpha_{11}\mathrm{R} + \mathrm{offsetChroma}. \quad (43)$$

[00166] Compared with the expression (26), an additional linear term α_{11}R is included in the expression (43). In the above expression (43), C_Pred represents a prediction chroma sample. R represents the downsampled residual luma sample. L_i represents a non-downsampled luma sample around the center luma sample (i.e., C), as shown in FIG. 11. In some implementations (e.g., on the decoder side), the non-downsampled luma sample L_i may represent a corresponding non-downsampled reconstructed luma sample.

[00167] In another example, the residual luma samples can be applied in a non-linear term in a filtering function of the residual based CCCM model. Like the reconstructed luma block, the residual luma block is also downsampled. The downsampled residual luma block has the same block size as that of the prediction chroma block. A filtering function of a residual based CCCM model when the residual luma samples and the luma samples are downsampled can be written as follows:

$$C_Pred = \alpha_0\mathrm{C} + \alpha_1\mathrm{N} + \alpha_2\mathrm{S} + \alpha_3\mathrm{E} + \alpha_4\mathrm{W} + \alpha_5\mathrm{P}_{NL} + \alpha_6\mathrm{B} + \alpha_7 \cdot R \cdot R. \quad (44)$$

[00168] In the expression (44) above, the linear term α_7R in the expression (42) is replaced by the non-linear term $\alpha_7 \cdot R \cdot R$. On the other hand, a filtering function of a residual based CCCM model when the luma samples are not downsample (but the residual luma samples are downsampled) can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \mathrm{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \mathrm{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \alpha_{11} \cdot R \cdot R + \mathrm{offsetChroma}. \quad (45)$$

[00169] In the expression (45) above, the linear term α_{11}R in the expression (43) is replaced by the non-linear term $\alpha_{11} \cdot R \cdot R$.

[00170] In yet another example, the residual luma samples can be directly applied in a linear term in a filtering function of the residual based CCCM model (e.g., without down-sampling the residual luma samples). That is, the residual luma block is not downsampled. A filtering function of a residual based two-parameter CCCM model when the luma samples are downsampled but the residual luma samples are not downsampled can be written as follows:

$$C_Pred = \alpha_0\mathrm{C} + \alpha_1\mathrm{N} + \alpha_2\mathrm{S} + \alpha_3\mathrm{E} + \alpha_4\mathrm{W} + \alpha_5\mathrm{P}_{NL} + \alpha_6\mathrm{B} + \sum_{i=0}^{5} \alpha_{i+7} \cdot R_i. \quad (46)$$

[00171] Compared with the expression (42), the linear term α_7R in the expression (42) is replaced by the linear term $\sum_{i=0}^{5} \alpha_{i+7} \cdot R_i$ in the expression (46). R_i represents a residual luma sample around the center luma sample C as shown in FIG. 16, with i = 0, 1, 2, 3, 4, or 5. On the other hand, a filtering function of a residual based CCCM model when the luma samples and the residual luma samples are not downsampled can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \mathrm{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \mathrm{offsetLuma})^2 + \beta) \gg$$

$$bitDepth) + \alpha_{10} \cdot \beta + \sum_{i=0}^{5} \alpha_{i+11} \cdot R_i + \text{offsetChroma}. \quad (47)$$

[00172] In the expression (47) above, the linear term α_{11}R in the expression (43) is replaced by the linear term $\sum_{i=0}^{5} \alpha_{i+11} \cdot R_i$.

[00173] In some implementations, the prediction model can be a luma-prediction based model. The processor may apply the prediction model to generate the prediction chroma samples based on the prediction luma samples. In some examples, the processor may apply the prediction model to generate the prediction chroma samples based on the prediction luma samples and the reconstructed luma samples. The prediction luma samples can be downsampled and used in a linear term in a filtering function of the prediction model. Alternatively, the prediction luma samples may be downsampled and used in a non-linear term in the filtering function of the prediction model. Alternatively, the prediction luma samples can be directly used in a linear term in the filtering function of the prediction model without down-sampling. The luma-prediction based model is described below with respect to three example cases.

[00174] In a first example case, the prediction model can be a prediction based CCLM model. The prediction luma samples may be used in the prediction based CCLM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the prediction based CCLM model to predict the chroma component.

[00175] For example, the prediction luma samples can be applied in a linear term in a filtering function of the prediction based CCLM model. Like the reconstructed luma block, the prediction luma block is also downsampled. The downsampled prediction luma block has the same block size as that of the prediction chroma block. The filtering function of the prediction based CCLM model can be written as follows:

$$\text{pred}_C(i,j) = \alpha \cdot \text{rec}_L'(i,j) + \beta \cdot \text{pred}_L'(i,j) + \gamma. \quad (48)$$

[00176] In the above expression (48), $\text{pred}_C(i,j)$ represents a prediction chroma sample in the video block; $\text{rec}_L'(i,j)$ and $\text{pred}_L'(i,j)$ represent a downsampled reconstructed luma sample and a downsampled prediction luma sample of the video block which are obtained by performing down-sampling on a reconstructed luma sample $\text{rec}_L(i,j)$ and a prediction luma sample $\text{pred}_L(i,j)$, respectively; and α, β, and γ represent linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma samples and downsampled prediction luma samples. The downsampled prediction luma sample $\text{pred}_L'(i,j)$ is applied in the linear term $\beta \cdot \text{pred}_L'(i,j)$ in the expression (48).

[00177] In another example, the prediction luma samples can be applied in a non-linear term in a filtering function of the prediction based CCLM model. Like the reconstructed luma block, the prediction luma block is also downsampled. The downsampled prediction luma block has the same block size as that of the prediction chroma block. The filtering function of the prediction based CCLM model can be written as follows:

[00178] $$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{rec_L}'(i,j) + \beta \cdot \mathrm{pred}'(i,j) \cdot \mathrm{pred_L}'(i,j) + \gamma. \quad (49)$$

[00179] In the above expression (49), $\mathrm{pred_C}(i,j)$ represents a prediction chroma sample in the video block; $\mathrm{rec_L}'(i,j)$ and $\mathrm{pred_L}'(i,j)$ represent a downsampled reconstructed luma sample and a downsampled prediction luma sample of the video block which are obtained by performing down-sampling on a reconstructed luma sample $\mathrm{rec_L}(i,j)$ and a prediction luma sample $\mathrm{pred_L}(i,j)$, respectively; and α, β, and γ represent linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma samples and downsampled prediction luma samples. The downsampled prediction luma sample $\mathrm{pred_L}'(i,j)$ is applied in the non-linear term $\beta \cdot \mathrm{pred_L}'(i,j) \cdot \mathrm{pred_L}'(i,j)$ in the expression (49).

[00180] In yet another example, the prediction luma samples can be directly applied in a linear term in a filtering function of the prediction based CCLM model (e.g., without down-sampling the prediction luma sample). That is, the prediction luma block is not downsampled, whereas the reconstructed luma block is downsampled. The filtering function of the prediction based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{rec_L}'(i,j) + \sum_{k=0}^{5} \alpha_k \cdot P_k(i,j) + \gamma. \quad (50)$$

[00181] In the above expression (50), $\mathrm{pred_C}(i,j)$ represents a prediction chroma sample in the video block, and $\mathrm{rec_L}'(i,j)$ represents a downsampled reconstructed luma sample which is obtained by performing down-sampling on the reconstructed luma samples $\mathrm{rec_L}(i,j)$. $P_k(i,j)$ represents a prediction luma sample of the video block as shown in FIG. 17, and α, α_k, and γ are linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma samples and corresponding prediction luma samples. The prediction luma sample $P_k(i,j)$ is applied in the linear term $\sum_{k=0}^{5} \alpha_k \cdot P_k(i,j)$ in the expression (50).

[00182] With reference to FIG. 17, "C" represents a center luma sample collocated with the chroma sample to be predicted, e.g., in a sample position (i,j) shown in the expression (50). P_k represents a prediction luma sample around the center luma sample, which is also referred to as $P_k(i,j)$ in the expression (50), with k = 0, 1, 2, 3, 4, or 5. P_k is non-downsampled.

[00183] In a second example case, the prediction model can be a prediction based GLM model. The prediction luma samples may be used in the prediction based GLM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the prediction based GLM model to predict the chroma component.

[00184] For example, the prediction luma samples can be applied in a linear term in a filtering function of the prediction based GLM model. Like the reconstructed luma block, the prediction luma block is also downsampled. The downsampled prediction luma block has the same block size as that of the prediction chroma block. A filtering function of a prediction based two-parameter GLM model and a filtering function of a prediction based three-parameter GLM model can be written in the following expressions (51) and (52), respectively.

$$C_Pred = \alpha \cdot G + \beta \cdot \mathrm{P} + \gamma, \qquad (51)$$

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \alpha_2 \cdot P + \alpha_3 \cdot \beta. \qquad (52)$$

[00185] In the above expressions (51) and (52), C_Pred represents a prediction chrome sample; and G, P, and L represent a luma sample gradient value, a downsampled prediction luma sample value, and a downsampled luma sample value, respectively. Compared with the expression (36), the linear term $\beta \cdot R$ in the expression (36) is replaced by the linear term $\beta \cdot \mathrm{P}$ in the expression (51). Compared with the expression (37), the linear term $\alpha_2 \cdot R$ in the expression (37) is replaced by the linear term $\alpha_2 \cdot \mathrm{P}$ in the expression (52).

[00186] In another example, the prediction luma samples can be applied in a non-linear term in a filtering function of the prediction based GLM model. Like the reconstructed luma block, the prediction luma block is also downsampled. The downsampled prediction luma block has the same block size as that of the prediction chroma block. A filtering function of a prediction based two-parameter GLM model and a filtering function of a prediction based three-parameter GLM model can be written in the following expressions (53) and (54), respectively:

$$C_Pred = \alpha \cdot G + \beta \cdot \mathrm{P} \cdot \mathrm{P} + \gamma, \qquad (53)$$

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \alpha_2 \cdot P \cdot P + \alpha_3 \cdot \beta. \qquad (54)$$

[00187] In the expression (53) above, the linear term $\beta \cdot P$ in the expression (51) is replaced by the non-linear term $\beta \cdot \mathrm{P} \cdot \mathrm{P}$. In the expression (54) above, the linear term $\alpha_2 \cdot P$ in the expression (52) is replaced by the linear term $\alpha_2 \cdot \mathrm{P} \cdot \mathrm{P}$.

[00188] In yet another example, the prediction luma samples can be directly applied in a linear term in a filtering function of the prediction based GLM model (e.g., without down-sampling the prediction luma samples). That is, the prediction luma block is not downsampled. A filtering function of a prediction based two-parameter GLM model and a filtering function

of a prediction based three-parameter GLM model can be written in the following expressions (55) and (56), respectively:

$$C_Pred = \alpha \cdot G + \sum_{i=0}^{5} \alpha_i \cdot P_i + \gamma, \qquad (55)$$

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \sum_{i=0}^{5} \alpha_i \cdot P_i + \alpha_6 \cdot \beta. \qquad (56)$$

[00189] In the above expressions (55) and (56), C_Pred represents a prediction chrome sample; and G, P_i, and L represent a luma sample gradient value, a prediction luma sample value (without down-sampling), and a downsampled luma sample value, respectively. P_i represents a prediction luma sample around the center luma sample C as shown in FIG. 17, with i = 0, 1, 2, 3, 4, or 5. P_i is not downsampled.

[00190] In a third example case, the prediction model can be a prediction based CCCM model. The prediction luma samples may be used in the prediction based CCCM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the prediction based CCCM model to predict the chroma component.

[00191] For example, the prediction luma samples can be applied in a linear term in a filtering function of the prediction based CCCM model. Like the reconstructed luma block, the prediction luma block is also downsampled. The downsampled prediction luma block has the same block size as that of the prediction chroma block. A filtering function of the prediction based CCCM model when the prediction luma samples and the luma samples are downsampled can be written as follows:

$$C_Pred = \alpha_0 \mathrm{C} + \alpha_1 \mathrm{N} + \alpha_2 \mathrm{S} + \alpha_3 \mathrm{E} + \alpha_4 \mathrm{W} + \alpha_5 \mathrm{P}_{NL} + \alpha_6 \mathrm{B} + \alpha_7 \cdot Pred. \qquad (57)$$

[00192] In the above expression, C_Pred represents a prediction chrome sample, and $Pred$ represents a downsampled prediction luma sample. Compared with the expression (42), the linear term $\alpha_7 \mathrm{R}$ in the expression (42) is replaced by the linear term $\alpha_7 \cdot Pred$ in the expression (57). On the other hand, a filtering function of a prediction based CCCM model when the luma samples are not downsampled (but the prediction luma samples are downsampled) can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \mathrm{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \mathrm{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \alpha_{11} \cdot Pred + \mathrm{offsetChroma}. \qquad (58)$$

[00193] In the expression (58), the linear term $\alpha_{11} \mathrm{R}$ in the expression (43) is replaced by the linear term $\alpha_{11} \cdot Pred$.

[00194] In another example, the prediction luma samples can be applied in a non-linear term in a filtering function of the prediction based CCCM model. Like the reconstructed luma block, the prediction luma block is also downsampled. The downsampled prediction luma block has

the same block size as that of the prediction chroma block. A filtering function of a prediction based CCCM model when the prediction luma samples and the luma samples are downsampled can be written as follows:

$$C_Pred = \alpha_0 \mathrm{C} + \alpha_1 \mathrm{N} + \alpha_2 \mathrm{S} + \alpha_3 \mathrm{E} + \alpha_4 \mathrm{W} + \alpha_5 \mathrm{P}_{NL} + \alpha_6 \mathrm{B} + \alpha_7 \cdot Pred \cdot Pred. \quad (59)$$

[00195] In the expression (59) above, the non-linear term $\alpha_7 \cdot R \cdot R$ in the expression (44) is replaced by the non-linear term $\alpha_7 \cdot Pred \cdot Pred$. On the other hand, a filtering function of a prediction based CCCM model when the luma samples are not downsampled (but the prediction luma samples are downsampled) can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \mathrm{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \mathrm{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \alpha_{11} \cdot Pred \cdot Pred + \mathrm{offsetChroma}. \quad (60)$$

[00196] In the expression (60) above, the non-linear term $\alpha_{11} \cdot R \cdot R$ in the expression (45) is replaced by the non-linear term $\alpha_{11} \cdot Pred \cdot Pred$.

[00197] In yet another example, the prediction luma samples can be directly applied in a linear term in a filtering function of the prediction based CCCM model (e.g., without down-sampling the prediction luma samples). That is, the prediction luma block is not downsampled. A filtering function of a prediction based two-parameter CCCM model (e.g., when the luma samples are downsampled but the prediction luma samples are not downsampled) can be written as follows:

$$C_Pred = \alpha_0 \mathrm{C} + \alpha_1 \mathrm{N} + \alpha_2 \mathrm{S} + \alpha_3 \mathrm{E} + \alpha_4 \mathrm{W} + \alpha_5 \mathrm{P}_{NL} + \alpha_6 \mathrm{B} + \sum_{i=0}^{5} \alpha_{i+7} \cdot Pred_i. \quad (61)$$

[00198] In the above expression (61), $Pred_i$ represents a prediction luma sample P_i as shown in FIG. 17. Compared with the expression (46), the non-linear term $\sum_{i=0}^{5} \alpha_{i+7} \cdot R_i$ in the expression (46) is replaced by the non-linear term $\sum_{i=0}^{5} \alpha_{i+7} \cdot Pred_i$ in the expression (61). On the other hand, a filtering function of a prediction based CCCM model when the luma samples and the residual luma samples are not downsampled can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \mathrm{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \mathrm{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \sum_{i=0}^{5} \alpha_{i+11} \cdot Pred_i + \mathrm{offsetChroma}. \quad (62)$$

[00199] In the expression (62), the linear term $\sum_{i=0}^{6} \alpha_{i+11} \cdot R_i$ in the expression (47) is replaced by the linear term $\sum_{i=0}^{5} \alpha_{i+11} \cdot Pred_i$.

[00200] In some implementations, the prediction model can be a prediction and residual based model. The processor may apply the prediction model to generate the prediction chroma samples based on the prediction luma samples and the residual luma samples. In some examples, the processor may apply the prediction model to generate the prediction chroma samples based on the prediction luma samples, the residual luma samples, and the reconstructed

luma samples. The prediction luma samples may be downsampled and used in a first linear term in a filtering function of the prediction model. The residual luma samples may be downsampled and used in a second linear term in the filtering function of the prediction model. Alternatively, the prediction luma samples may be downsampled and used in a first non-linear term in the filtering function of the prediction model. The residual luma samples may be downsampled and used in a second non-linear term in the filtering function of the prediction model. The prediction and residual based model is described below with respect to three example cases.

[00201] In a first example case, the prediction model can be a prediction and residual based CCLM model. The prediction luma samples and the residual luma samples may be used in the model to predict the chroma component. Further, the reconstructed luma samples may also be used in the model to predict the chroma component.

[00202] For example, the prediction luma samples can be applied in a first linear term and the residual luma samples can be applied in a second linear term in a filtering function of the prediction and residual based CCLM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the same block size as that of the prediction chroma block. The filtering function of the prediction and residual based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha_0 \cdot \mathrm{rec_L}'(i,j) + \alpha_1 \cdot \mathrm{pred_L}'(i,j) + \alpha_2 \cdot \mathrm{res_L}'(i,j) + \beta. \quad (63)$$

[00203] In the above expression (63), $\mathrm{pred_C}(i,j)$ represents a prediction chroma sample in the video block; $\mathrm{rec_L}'(i,j)$, $\mathrm{pred_L}'(i,j)$, and $\mathrm{res_L}'(i,j)$ represent a downsampled reconstructed luma sample, a downsampled prediction luma sample, and a downsampled residual luma sample of the video block which are obtained by performing down-sampling on a reconstructed luma sample $\mathrm{rec_L}(i,j)$, a prediction luma sample $\mathrm{pred_L}(i,j)$, and a residual luma sample $\mathrm{res_L}(i,j)$, respectively; and α_i, β, and γ represent linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled reconstructed luma samples, downsampled prediction luma samples, and downsampled residual luma samples.

[00204] In another example, the prediction luma samples can be applied in a first non-linear term and the residual luma samples can be applied in a second non-linear term in a filtering function of the prediction and residual based CCLM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the

same block size as that of the prediction chroma block. The filtering function of the prediction and residual based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha_0 \cdot \mathrm{rec_L}'(i,j) + \alpha_1 \cdot \mathrm{pred_L}'(i,j) \cdot \mathrm{pred_L}'(i,j) + \alpha_2 \cdot \mathrm{res_L}'(i,j) \cdot \mathrm{res_L}'(i,j) + \beta. \quad (64)$$

[00205] In the expression (64) above, the linear terms $\alpha_1 \cdot \mathrm{pred_L}'(i,j)$ and $\alpha_2 \cdot \mathrm{res_L}'(i,j)$ in the expression (63) are replaced by the non-linear terms $\alpha_1 \cdot \mathrm{pred_L}'(i,j) \cdot \mathrm{pred_L}'(i,j)$ and $\alpha_2 \cdot \mathrm{res_L}'(i,j) \cdot \mathrm{res_L}'(i,j)$, respectively.

[00206] In yet another example, the prediction luma samples can be applied in a first linear term and the residual luma samples can be applied in a second linear term in a filtering function of the prediction and residual based CCLM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the same block size as that of the prediction chroma block. The filtering function of the prediction and residual based CCLM model can be written as follows:

$$\mathrm{pred_C}(i,j) = \alpha \cdot \mathrm{res_L}'(i,j) + \beta \cdot \mathrm{pred}'_L(i,j) + \gamma. \quad (65)$$

[00207] In the above expression (65), α, β, and γ represent linear model parameters which are derived from neighboring chroma samples and their corresponding downsampled prediction luma samples and downsampled residual luma samples.

[00208] In a second example case, the prediction model can be a prediction and residual based GLM model. The prediction luma samples and the residual luma samples may be used in the prediction and residual based GLM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the prediction and residual based GLM model to predict the chroma component.

[00209] For example, the prediction luma samples can be applied in a first linear term and the residual luma samples can be applied in a second linear term in a filtering function of the prediction and residual based GLM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the same block size as that of the prediction chroma block. The filtering function of the prediction and residual based GLM model can be written as follows:

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot \mathrm{P} + \alpha_2 \cdot R + \beta. \quad (66)$$

[00210] In the above expression, C_Pred, G, P, and R represent a prediction chroma sample, a luma sample gradient, a downsampled prediction luma sample, and a downsampled residual

luma sample, respectively.

[00211] In another example, the prediction luma samples can be applied in a first non-linear term and the residual luma samples can be applied in a second non-linear term in a filtering function of the prediction and residual based GLM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the same block size as that of the prediction chroma block. A filtering function of a prediction and residual based two-parameter GLM model can be written as shown in expression (67):

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot \mathrm{P} \cdot \mathrm{P} + \alpha_2 \cdot R \cdot R + \beta. \qquad (67)$$

[00212] A filtering function of a prediction and residual based three-parameter GLM model can be written as shown in expression (68):

$$C_Pred = \alpha_0 \cdot G + \alpha_1 \cdot L + \alpha_2 \cdot P \cdot \mathrm{P} + \alpha_3 \cdot R \cdot \mathrm{R} + \alpha_4 \cdot \beta. \qquad (68)$$

[00213] In a third example case, the prediction model can be a prediction and residual based CCCM model. The prediction luma samples and the residual luma samples may be used in the prediction and residual based CCCM model to predict the chroma component. Further, the reconstructed luma samples may also be used in the prediction and residual based CCCM model to predict the chroma component.

[00214] For example, the prediction luma samples can be applied in a first linear term and the residual luma samples can be applied in a second linear term in a filtering function of the prediction and residual based CCCM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the same block size as that of the prediction chroma block. A filtering function of the prediction and residual based CCCM model (when the luma samples, the prediction luma samples, and the residual luma samples are downsampled) can be written as follows:

$$C_Pred = \alpha_0\mathrm{C} + \alpha_1\mathrm{N} + \alpha_2\mathrm{S} + \alpha_3\mathrm{E} + \alpha_4\mathrm{W} + \alpha_5\mathrm{P}_{NL} + \alpha_6\mathrm{B} + \alpha_7 \cdot Pred + \alpha_8 \cdot R. \qquad (69)$$

[00215] In the above expression (69), C_Pred represents a prediction chroma sample, and $Pred$ represents a downsampled prediction luma sample. R represents a downsampled residual luma sample. Compared with the expression (57), an additional linear term $\alpha_8 \cdot R$ is added to the expression (69). On the other hand, a filtering function of a prediction based CCCM model (when the luma samples are not downsampled but the residual luma samples and the prediction luma samples are downsampled) can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \text{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \text{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \alpha_{11} \cdot Pred + \alpha_{12} \cdot R + \text{offsetChroma}. \quad (70)$$

[00216] Compared with the expression (58), an additional linear term $\alpha_{12} \cdot R$ is added to the expression (70).

[00217] In another example, the prediction luma samples can be applied in a first non-linear term and the residual luma samples can be applied in a second non-linear term in a filtering function of the prediction and residual based CCCM model. Like the reconstructed luma block, the prediction luma block and the residual luma block may also be downsampled. The downsampled prediction luma block and the downsampled residual luma block may have the same block size as that of the prediction chroma block. A filtering function of the prediction and residual based CCCM model (when the luma samples, the prediction luma samples, and the residual luma samples are downsampled) can be written as follows:

$$C_Pred = \alpha_0 \mathrm{C} + \alpha_1 \mathrm{N} + \alpha_2 \mathrm{S} + \alpha_3 \mathrm{E} + \alpha_4 \mathrm{W} + \alpha_5 \mathrm{P} + \alpha_6 \mathrm{B} + \alpha_7 \cdot Pred \cdot Pred + \alpha_8 \cdot R \cdot R. \quad (71)$$

[00218] Compared with the expression (59), an additional non-linear term $\alpha_8 \cdot R \cdot R$ is added to the expression (71). On the other hand, a filtering function of a prediction and residual based CCCM model when the luma samples are not downsampled (but the residual luma samples and the prediction luma samples are downsampled) can be written as follows:

$$C_Pred = \sum_{i=0}^{5} \alpha_i \cdot (L_i - \text{offsetLuma}) + \sum_{i=6}^{9} \alpha_i \cdot (((L_{i-4} - \text{offsetLuma})^2 + \beta) \gg bitDepth) + \alpha_{10} \cdot \beta + \alpha_{11} \cdot Pred \cdot Pred + \alpha_{12} \cdot R \cdot R + \text{offsetChroma}. \quad (72)$$

[00219] Compared with the expression (60), an additional non-linear term $\alpha_{12} \cdot R \cdot R$ is added to the expression (72).

[00220] FIG. 18 shows a computing environment 1800 coupled with a user interface 1850. The computing environment 1800 can be part of a data processing server. The computing environment 1800 includes a processor 1820, a memory 1830, and an Input/Output (I/O) interface 1840.

[00221] The processor 1820 typically controls overall operations of the computing environment 1800, such as the operations associated with display, data acquisition, data communications, and image processing. The processor 1820 may include one or more processors to execute instructions to perform all or some of the steps in the above-described methods. Moreover, the processor 1820 may include one or more modules that facilitate the interaction between the processor 1820 and other components. The processor may be a Central Processing Unit (CPU), a microprocessor, a single chip machine, a Graphical Processing Unit

(GPU), or the like.

[00222] The memory 1830 is configured to store various types of data to support the operation of the computing environment 1800. The memory 1830 may include predetermined software 1832. Examples of such data includes instructions for any applications or methods operated on the computing environment 1800, video datasets, image data, etc. The memory 1830 may be implemented by using any type of volatile or non-volatile memory devices, or a combination thereof, such as a Static Random Access Memory (SRAM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), an Erasable Programmable Read-Only Memory (EPROM), a Programmable Read-Only Memory (PROM), a Read-Only Memory (ROM), a magnetic memory, a flash memory, a magnetic or optical disk.

[00223] The I/O interface 1840 provides an interface between the processor 1820 and peripheral interface modules, such as a keyboard, a click wheel, buttons, and the like. The buttons may include but are not limited to, a home button, a start scan button, and a stop scan button. The I/O interface 1840 can be coupled with an encoder or decoder.

[00224] In an embodiment, there is also provided a non-transitory computer-readable storage medium comprising a plurality of programs, for example, in the memory 1830, executable by the processor 1820 in the computing environment 1800, for performing the above-described methods and/or storing a bitstream generated by the encoding method described above or a bitstream to be decoded by the decoding method described above. In one example, the plurality of programs may be executed by the processor 1820 in the computing environment 1800 to receive (for example, from the video encoder 20 in FIG. 2) a bitstream or data stream including encoded video information (for example, video blocks representing encoded video frames, and/or associated one or more syntax elements, etc.), and may also be executed by the processor 1820 in the computing environment 1800 to perform the decoding method described above according to the received bitstream or data stream. In another example, the plurality of programs may be executed by the processor 1820 in the computing environment 1800 to perform the encoding method described above to encode video information (for example, video blocks representing video frames, and/or associated one or more syntax elements, etc.) into a bitstream or data stream, and may also be executed by the processor 1820 in the computing environment 1800 to transmit the bitstream or data stream (for example, to the video decoder 30 in FIG. 3). Alternatively, the non-transitory computer-readable storage medium may have stored therein a bitstream or a data stream comprising encoded video information (for example, video blocks representing encoded video frames, and/or associated one or more syntax elements etc.) generated by an encoder (for example, the video encoder 20

in FIG. 2) using, for example, the encoding method described above for use by a decoder (for example, the video decoder 30 in FIG. 3) in decoding video data. The non-transitory computer-readable storage medium may be, for example, a ROM, a Random Access Memory (RAM), a CD-ROM, a magnetic tape, a floppy disc, an optical data storage device or the like.

[00225] In an embodiment, there is provided a bitstream generated by the encoding method described above or a bitstream to be decoded by the decoding method described above. In an embodiment, there is provided a bitstream comprising encoded video information generated by the encoding method described above or encoded video information to be decoded by the decoding method described above.

[00226] In an embodiment, the is also provided a computing device comprising one or more processors (for example, the processor 1820); and the non-transitory computer-readable storage medium or the memory 1830 having stored therein a plurality of programs executable by the one or more processors, wherein the one or more processors, upon execution of the plurality of programs, are configured to perform the above-described methods.

[00227] In an embodiment, there is also provided a computer program product having instructions for storage or transmission of a bitstream comprising encoded video information generated by the encoding method described above or encoded video information to be decoded by the decoding method described above. In an embodiment, there is also provided a computer program product comprising a plurality of programs, for example, in the memory 1830, executable by the processor 1820 in the computing environment 1800, for performing the above-described methods. For example, the computer program product may include the non-transitory computer-readable storage medium.

[00228] In an embodiment, the computing environment 1800 may be implemented with one or more ASICs, DSPs, Digital Signal Processing Devices (DSPDs), Programmable Logic Devices (PLDs), FPGAs, GPUs, controllers, micro-controllers, microprocessors, or other electronic components, for performing the above methods.

[00229] In an embodiment, there is also provided a method of storing a bitstream, comprising storing the bitstream on a digital storage medium, wherein the bitstream comprises encoded video information generated by the encoding method described above or encoded video information to be decoded by the decoding method described above.

[00230] In an embodiment, there is also provided a method for transmitting a bitstream generated by the encoder described above. In an embodiment, there is also provided a method for receiving a bitstream to be decoded by the decoder described above.

[00231] The description of the present disclosure has been presented for purposes of

illustration and is not intended to be exhaustive or limited to the present disclosure. Many modifications, variations, and alternative implementations will be apparent to those of ordinary skill in the art having the benefit of the teachings presented in the foregoing descriptions and the associated drawings.

[00232] Unless specifically stated otherwise, an order of steps of the method according to the present disclosure is only intended to be illustrative, and the steps of the method according to the present disclosure are not limited to the order specifically described above, but may be changed according to practical conditions. In addition, at least one of the steps of the method according to the present disclosure may be adjusted, combined or deleted according to practical requirements.

[00233] The examples were chosen and described in order to explain the principles of the disclosure and to enable others skilled in the art to understand the disclosure for various implementations and to best utilize the underlying principles and various implementations with various modifications as are suited to the particular use contemplated. Therefore, it is to be understood that the scope of the disclosure is not to be limited to the specific examples of the implementations disclosed and that modifications and other implementations are intended to be included within the scope of the present disclosure.

CLAIMS

What is claimed is:

1. A method for video decoding, comprising:

determining, by a decoder, at least one of residual luma sample information or prediction luma sample information associated with a video block from a video frame of a video; and

applying, by the decoder, a prediction model to generate prediction chroma samples associated with the video block based on the at least one of the residual luma sample information or the prediction luma sample information.

2. The method of claim 1, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block comprises determining residual luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples comprises applying the prediction model to generate the prediction chroma samples based on the residual luma samples.

3. The method of claim 2, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block further comprises determining prediction luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples further comprises:

generating reconstructed luma samples associated with the video block based on the residual luma samples and the prediction luma samples; and

applying the prediction model to generate the prediction chroma samples based on the residual luma samples and the reconstructed luma samples.

4. The method of claim 2, wherein the residual luma samples are downsampled and used in a linear term in a filtering function of the prediction model.

5. The method of claim 2, wherein the residual luma samples are downsampled and

used in a non-linear term in a filtering function of the prediction model.

6. The method of claim 2, wherein the residual luma samples are directly used in a linear term in a filtering function of the prediction model without down-sampling.

7. The method of claim 1, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block comprises determining prediction luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples comprises applying the prediction model to generate the prediction chroma samples based on the prediction luma samples.

8. The method of claim 7, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block further comprises determining residual luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples further comprises:

generating reconstructed luma samples associated with the video block based on the residual luma samples and the prediction luma samples; and

applying the prediction model to generate the prediction chroma samples based on the prediction luma samples and the reconstructed luma samples.

9. The method of claim 7, wherein the prediction luma samples are downsampled and used in a linear term in a filtering function of the prediction model.

10. The method of claim 7, wherein the prediction luma samples are downsampled and used in a non-linear term in a filtering function of the prediction model.

11. The method of claim 7, wherein the prediction luma samples are directly used in a linear term in a filtering function of the prediction model without down-sampling.

12. The method of claim 1, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block comprises determining residual luma samples and prediction luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples comprises applying the prediction model to generate the prediction chroma samples based on the residual luma samples and the prediction luma samples.

13. The method of claim 12, wherein applying the prediction model to generate the prediction chroma samples further comprises:

generating reconstructed luma samples associated with the video block based on the residual luma samples and the prediction luma samples; and

applying the prediction model to generate the prediction chroma samples based on the residual luma samples, the prediction luma samples, and the reconstructed luma samples.

14. The method of claim 12, wherein:

the prediction luma samples are downsampled and used in a first linear term in a filtering function of the prediction model; and

the residual luma samples are downsampled and used in a second linear term in the filtering function of the prediction model.

15. The method of claim 12, wherein:

the prediction luma samples are downsampled and used in a first non-linear term in a filtering function of the prediction model; and

the residual luma samples are downsampled and used in a second non-linear term in the filtering function of the prediction model.

16. The method of claim 1, wherein the prediction model is a cross component linear model (CCLM), a gradient linear model (GLM), or a convolutional cross component model (CCCM).

17. A method for video encoding, comprising:

determining, by an encoder, at least one of residual luma sample information or prediction luma sample information associated with a video block from a video frame of a video; and

applying, by the encoder, a prediction model to generate prediction chroma samples associated with the video block based on the at least one of the residual luma sample information or the prediction luma sample information.

18. The method of claim 17, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block comprises determining residual luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples comprises applying the prediction model to generate the prediction chroma samples based on the residual luma samples.

19. The method of claim 18, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block further comprises determining prediction luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples further comprises:

generating reconstructed luma samples associated with the video block based on the residual luma samples and the prediction luma samples; and

applying the prediction model to generate the prediction chroma samples based on the residual luma samples and the reconstructed luma samples.

20. The method of claim 18, wherein the residual luma samples are downsampled and used in a linear term in a filtering function of the prediction model.

21. The method of claim 18, wherein the residual luma samples are downsampled and used in a non-linear term in a filtering function of the prediction model.

22. The method of claim 18, wherein the residual luma samples are directly used in a linear term in a filtering function of the prediction model without down-sampling.

23. The method of claim 17, wherein:

determining at least one of the residual luma sample information or the prediction luma

sample information associated with the video block comprises determining prediction luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples comprises applying the prediction model to generate the prediction chroma samples based on the prediction luma samples.

24. The method of claim 23, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block further comprises determining residual luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples further comprises:

generating reconstructed luma samples associated with the video block based on the residual luma samples and the prediction luma samples; and

applying the prediction model to generate the prediction chroma samples based on the prediction luma samples and the reconstructed luma samples.

25. The method of claim 23, wherein the prediction luma samples are downsampled and used in a linear term in a filtering function of the prediction model.

26. The method of claim 23, wherein the prediction luma samples are downsampled and used in a non-linear term in a filtering function of the prediction model.

27. The method of claim 23, wherein the prediction luma samples are directly used in a linear term in a filtering function of the prediction model without down-sampling.

28. The method of claim 17, wherein:

determining at least one of the residual luma sample information or the prediction luma sample information associated with the video block comprises determining residual luma samples and prediction luma samples associated with the video block; and

applying the prediction model to generate the prediction chroma samples comprises applying the prediction model to generate the prediction chroma samples based on the residual luma samples and the prediction luma samples.

29. The method of claim 28, wherein applying the prediction model to generate the prediction chroma samples further comprises:

generating reconstructed luma samples associated with the video block based on the residual luma samples and the prediction luma samples; and

applying the prediction model to generate the prediction chroma samples based on the residual luma samples, the prediction luma samples, and the reconstructed luma samples.

30. The method of claim 28, wherein:

the prediction luma samples are downsampled and used in a first linear term in a filtering function of the prediction model; and

the residual luma samples are downsampled and used in a second linear term in the filtering function of the prediction model.

31. The method of claim 28, wherein:

the prediction luma samples are downsampled and used in a first non-linear term in a filtering function of the prediction model; and

the residual luma samples are downsampled and used in a second non-linear term in the filtering function of the prediction model.

32. The method of claim 17, wherein the prediction model is a cross component linear model (CCLM), a gradient linear model (GLM), or a convolutional cross component model (CCCM).

33. A method at a decoder side, comprising:

receiving a bitstream by a decoder, wherein the bitstream is decoded by performing the method of video decoding according to any one of claims 1-16.

34. A method at an encoder side, comprising:

storing a bitstream by an encoder, wherein the bitstream is generated by performing the method of video encoding according to any one of claims 17-32.

35. An apparatus for video decoding, comprising:

a memory configured to store a bitstream; and

a processor coupled to the memory and configured to perform the method according to

any one of claims 1-16 to decode the bitstream or the method according to claim 33 to receive the bitstream.

36. An apparatus for video encoding, comprising:

a memory configured to store a bitstream; and

a processor coupled to the memory and configured to perform the method according to any one of claims 17-32 to generate the bitstream or the method according to claim 34 to store the bitstream.

37. A non-transitory computer-readable storage medium having stored therein a bitstream, wherein the bitstream is decoded by the method according to any one of claims 1-16 or the bitstream is received by the method according to claim 33.

38. A non-transitory computer-readable storage medium having stored therein a bitstream, wherein the bitstream is generated by the method according to any one of claims 17-32 or the bitstream is stored by the method according to claim 34.

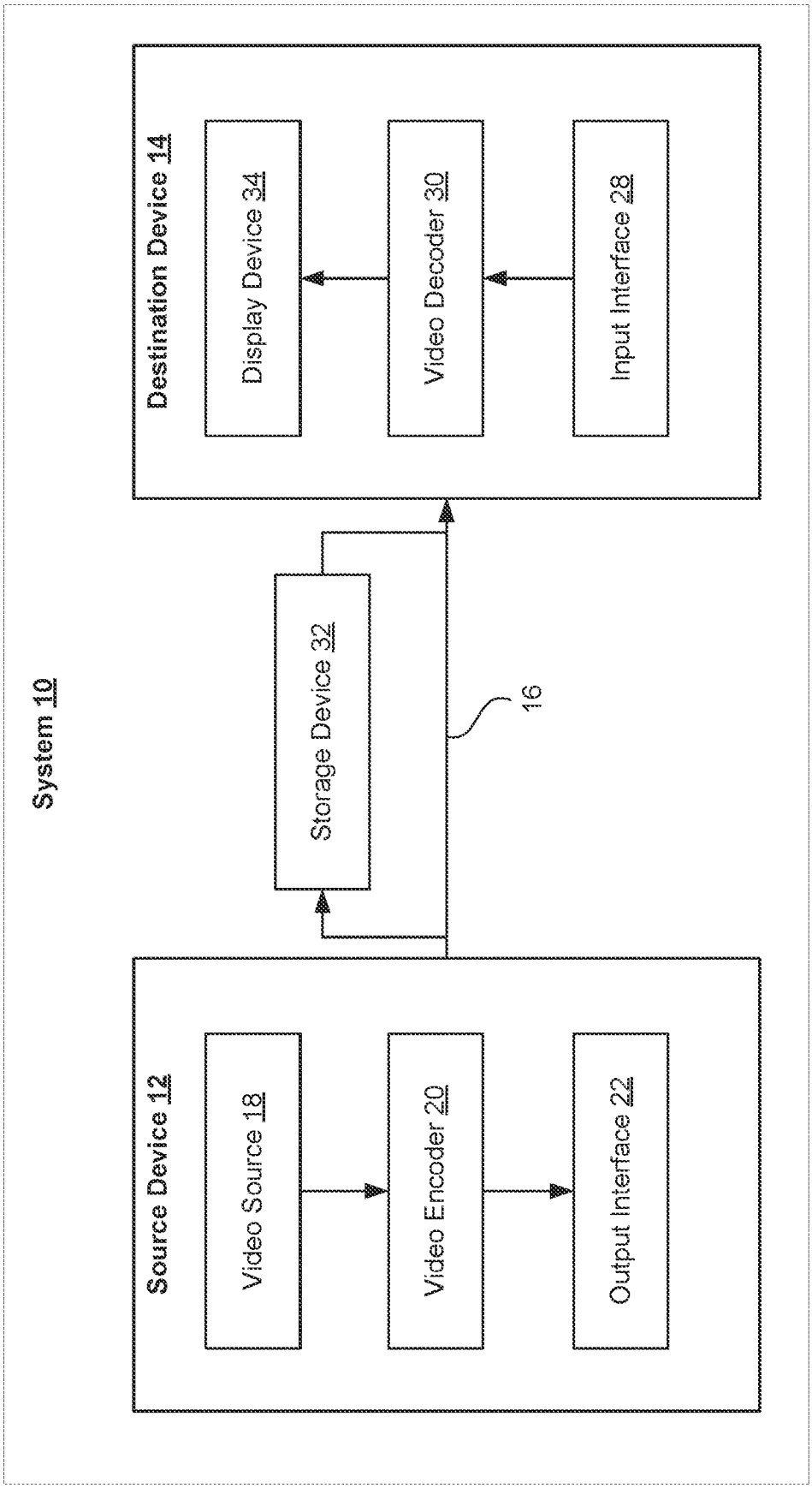

System 10
Source Device 12
Video Source 18
Video Encoder 20
Output Interface 22
Storage Device 32
16
Destination Device 14
Display Device 34
Video Decoder 30
Input Interface 28

FIG. 1

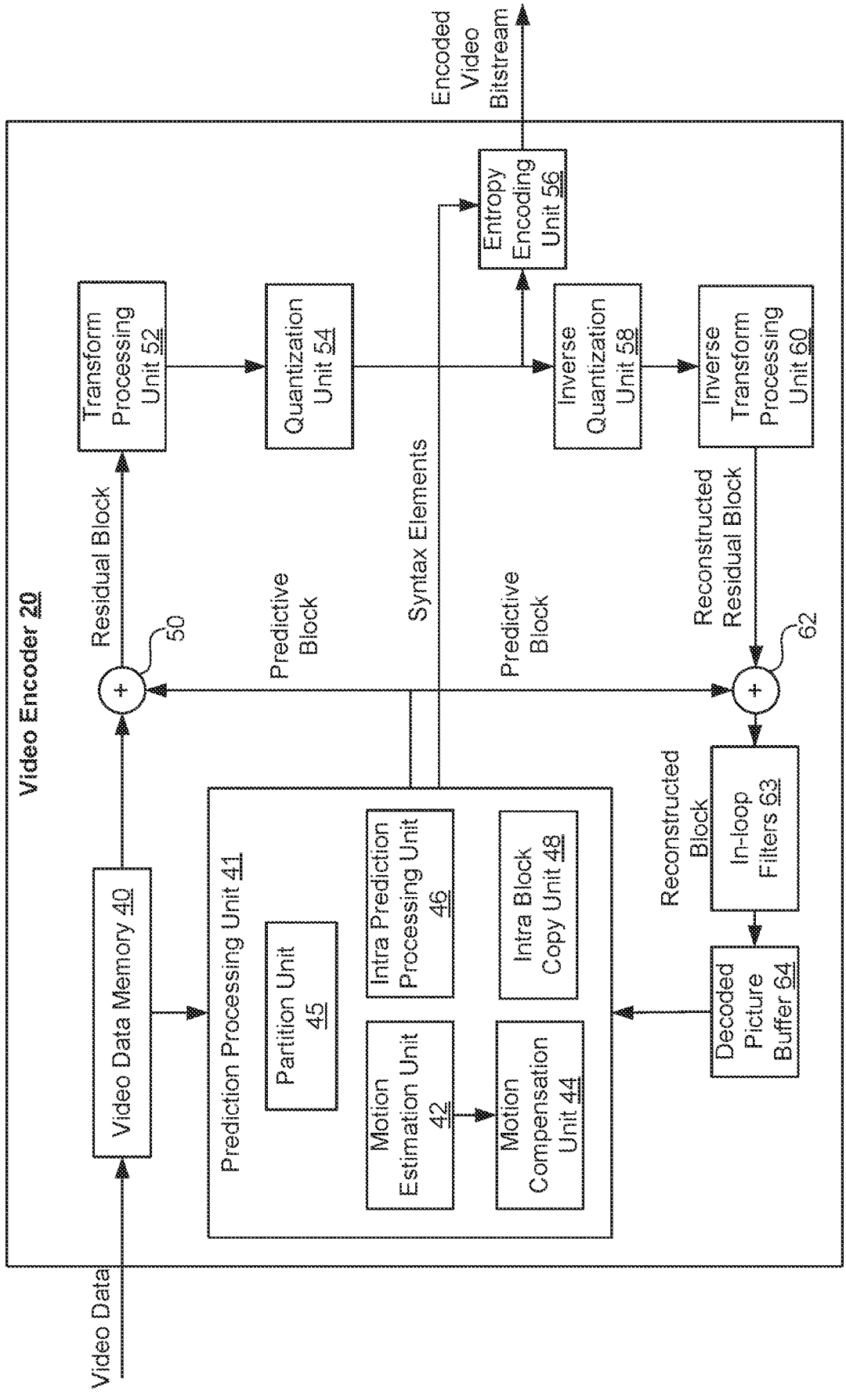

Video Encoder 20
Video Data
Video Data Memory 40
Prediction Processing Unit 41
Partition Unit 45
Motion Estimation Unit 42
Motion Compensation Unit 44
Intra Prediction Processing Unit 46
Intra Block Copy Unit 48
Residual Block
50
Transform Processing Unit 52
Quantization Unit 54
Entropy Encoding Unit 56
Encoded Video Bitstream
Syntax Elements
Predictive Block
Predictive Block
Inverse Quantization Unit 58
Inverse Transform Processing Unit 60
Reconstructed Residual Block
62
Reconstructed Block
In-loop Filters 63
Decoded Picture Buffer 64

FIG. 2

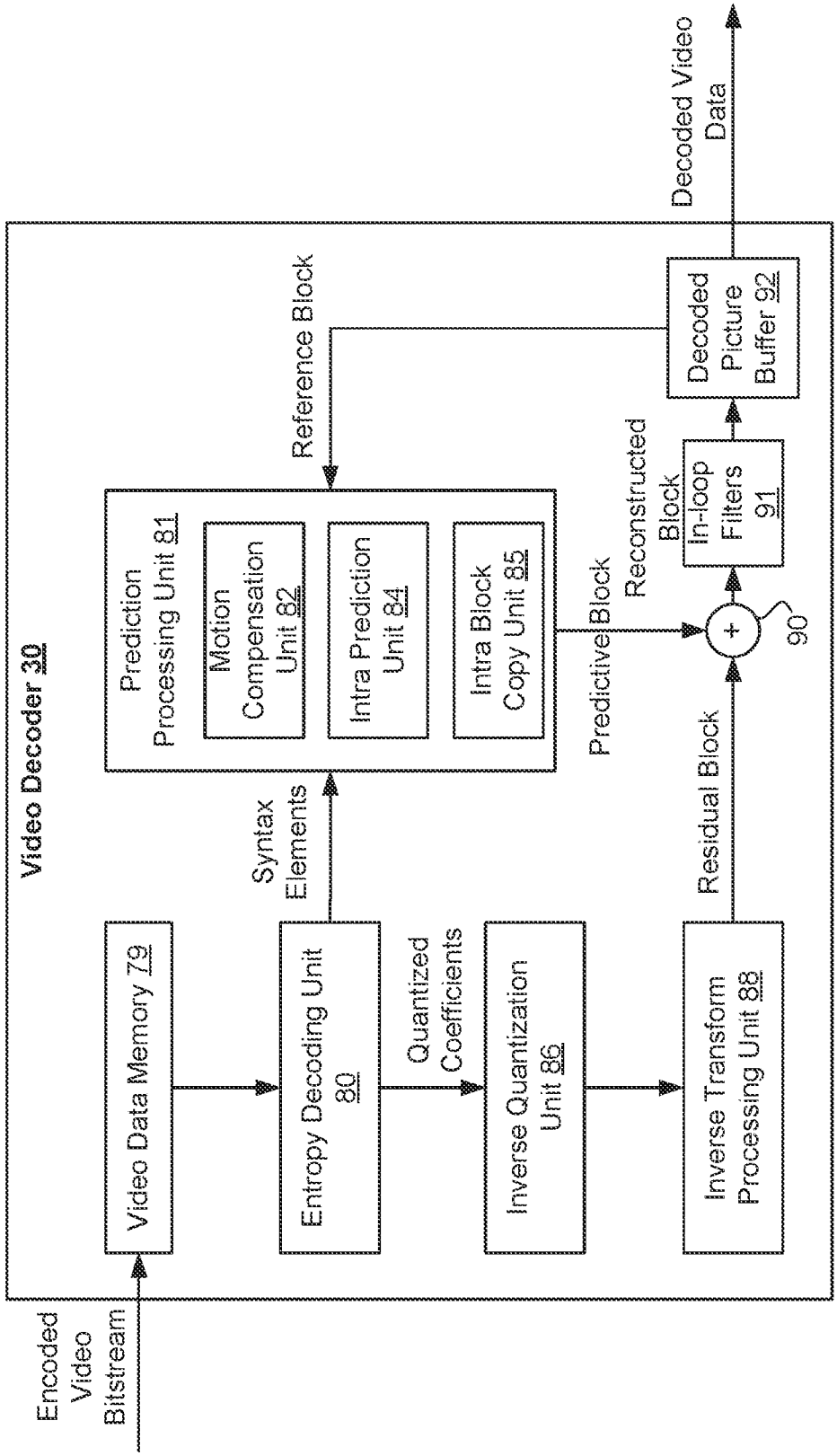

Video Decoder 30
Video Data Memory 79
Entropy Decoding Unit 80
Inverse Quantization Unit 86
Inverse Transform Processing Unit 88
Prediction Processing Unit 81
Motion Compensation Unit 82
Intra Prediction Unit 84
Intra Block Copy Unit 85
In-loop Filters 91
Decoded Picture Buffer 92
Encoded Video Bitstream
Syntax Elements
Quantized Coefficients
Residual Block
Predictive Block
Reconstructed Block
Reference Block
90
Decoded Video Data

FIG. 3

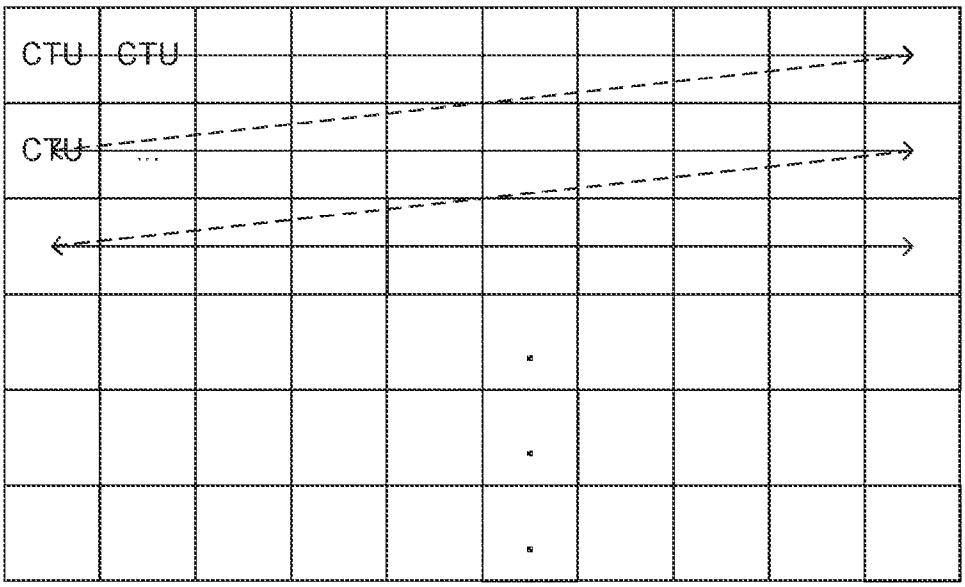

CTU
CTU
CTU

FIG. 4A

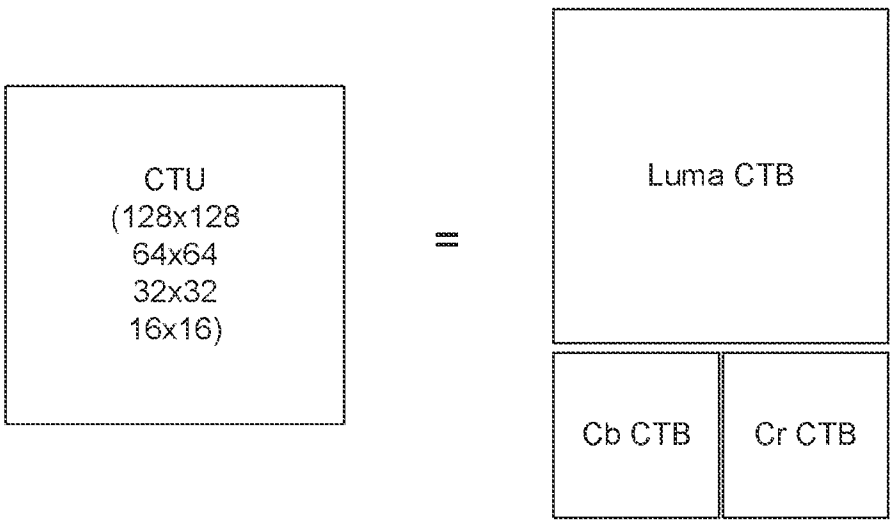

CTU
(128x128
64x64
32x32
16x16)
=
Luma CTB
Cb CTB
Cr CTB

FIG. 4B

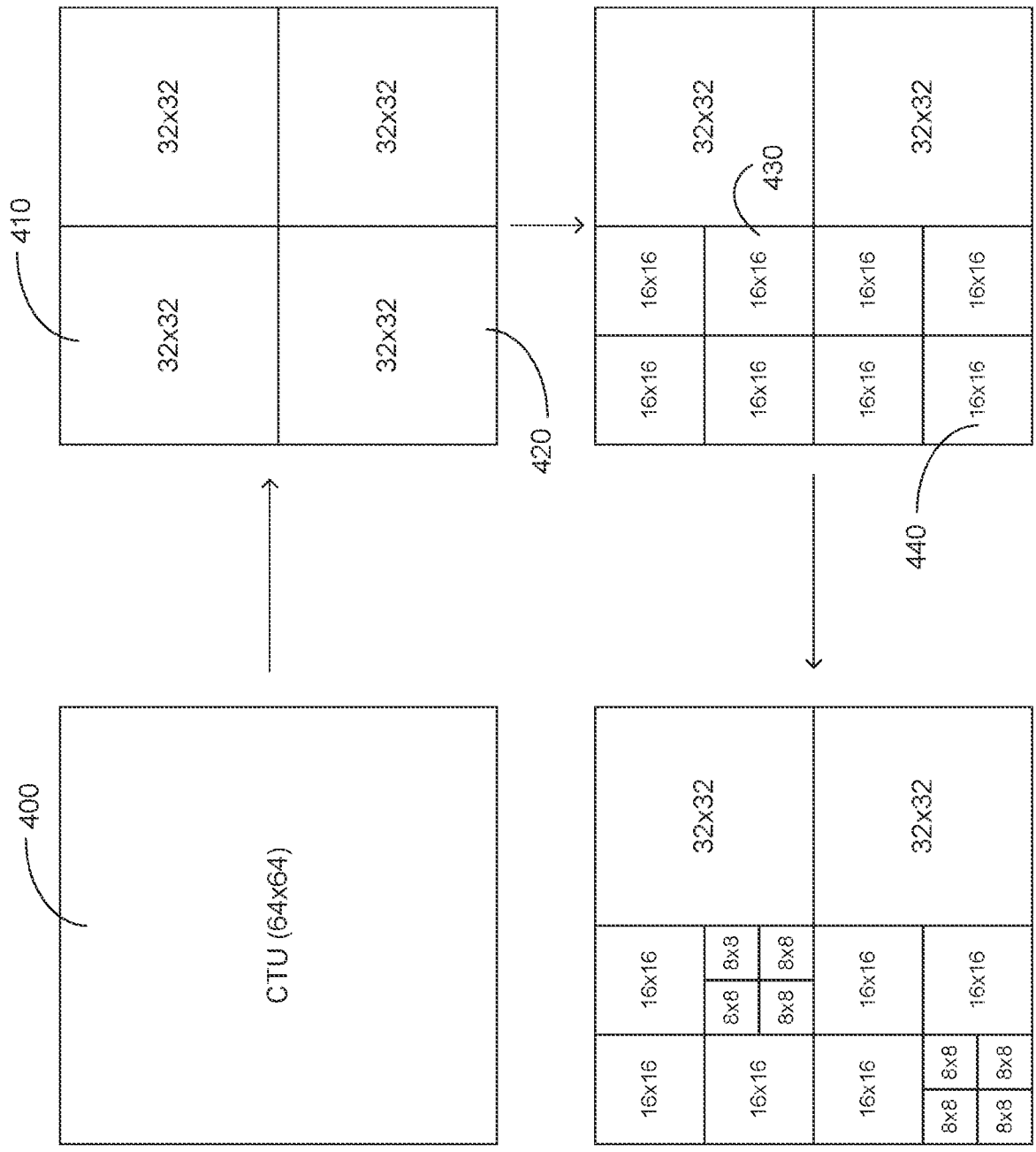

400
CTU (64x64)
410
32x32
32x32
32x32
32x32
420
430
16x16
8x8
440

FIG. 4C

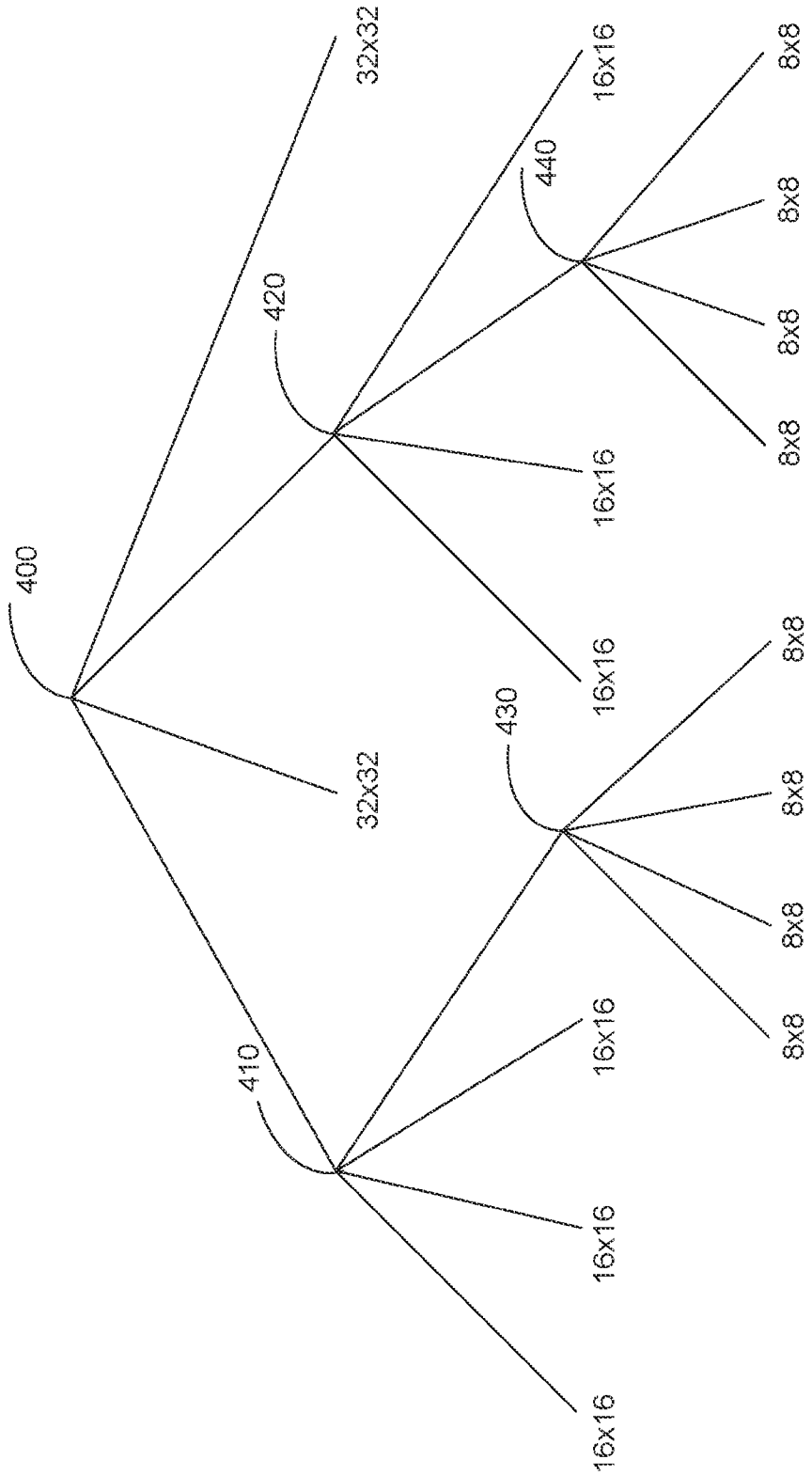

400
32x32
420
16x16
440
8x8
8x8
8x8
16x16
16x16
32x32
430
8x8
8x8
8x8
410
16x16
16x16
16x16

FIG. 4D

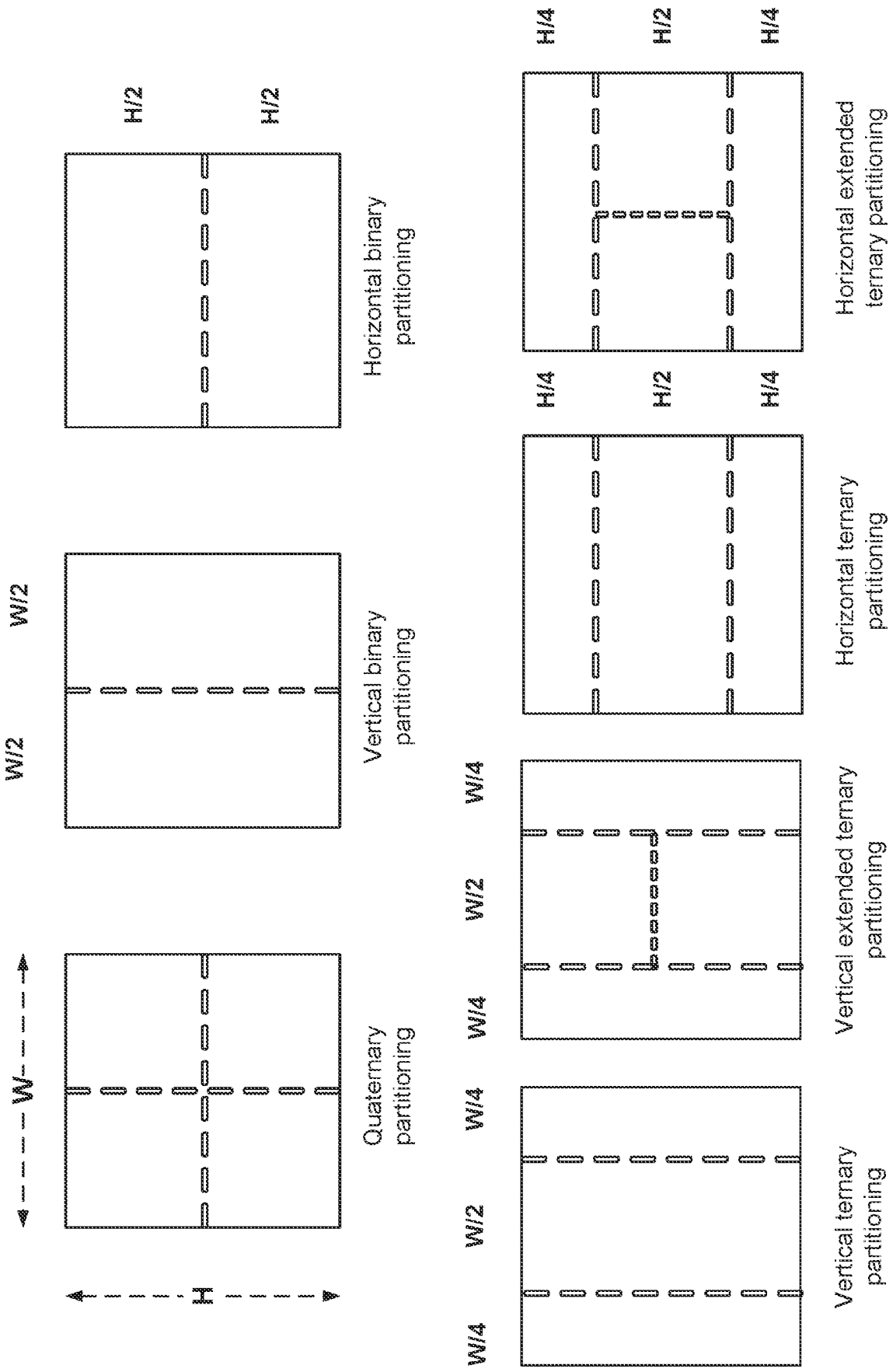

W
H
W/2
W/2
H/2
H/2
Quaternary partitioning
Vertical binary partitioning
Horizontal binary partitioning
W/4
W/2
W/4
W/4
W/2
W/4
H/4
H/2
H/4
H/4
H/2
H/4
Vertical ternary partitioning
Vertical extended ternary partitioning
Horizontal ternary partitioning
Horizontal extended ternary partitioning

FIG. 4E

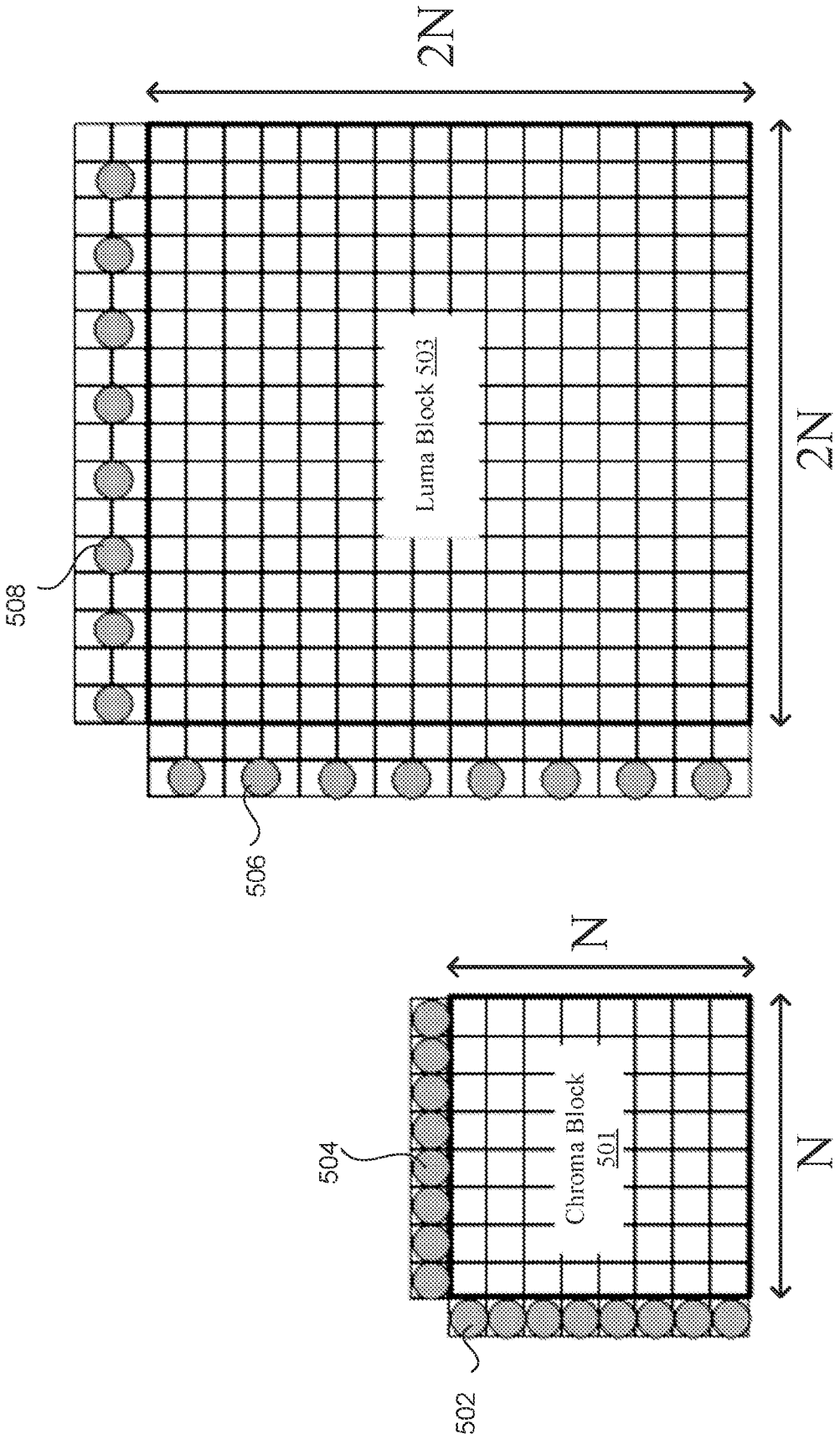


FIG. 5

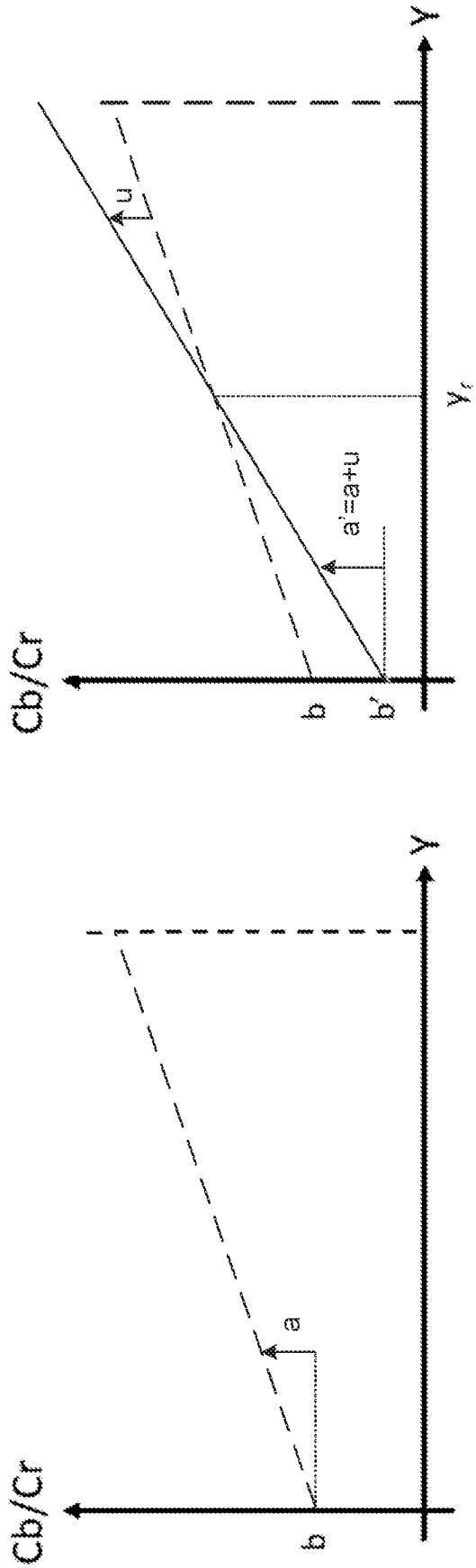


FIG. 6

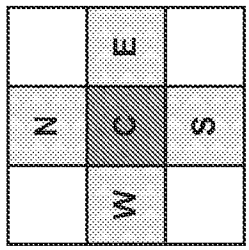


FIG. 7

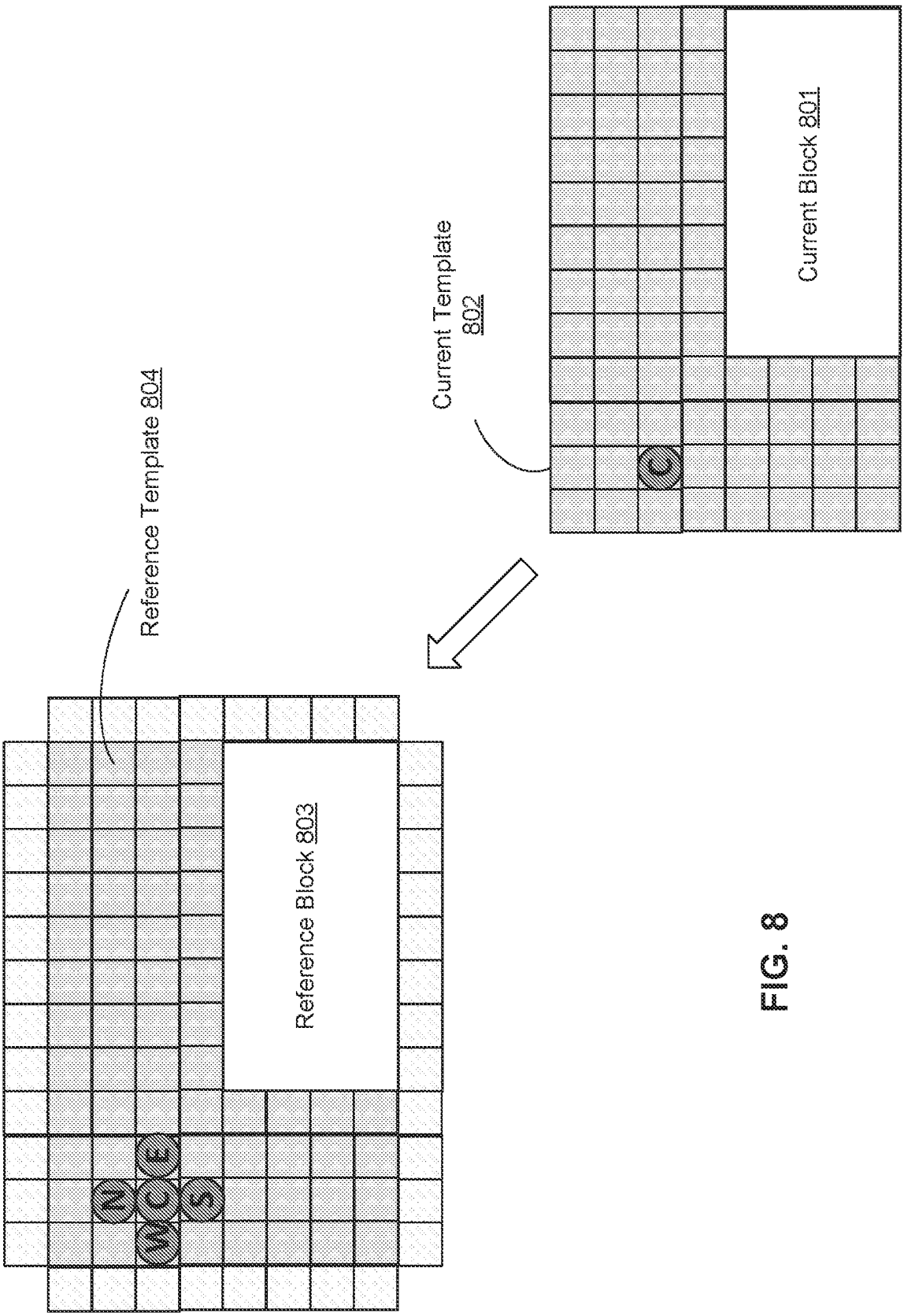

Current Block 801
Current Template 802
C
Reference Template 804
Reference Block 803
N
W
C
E
S

FIG. 8

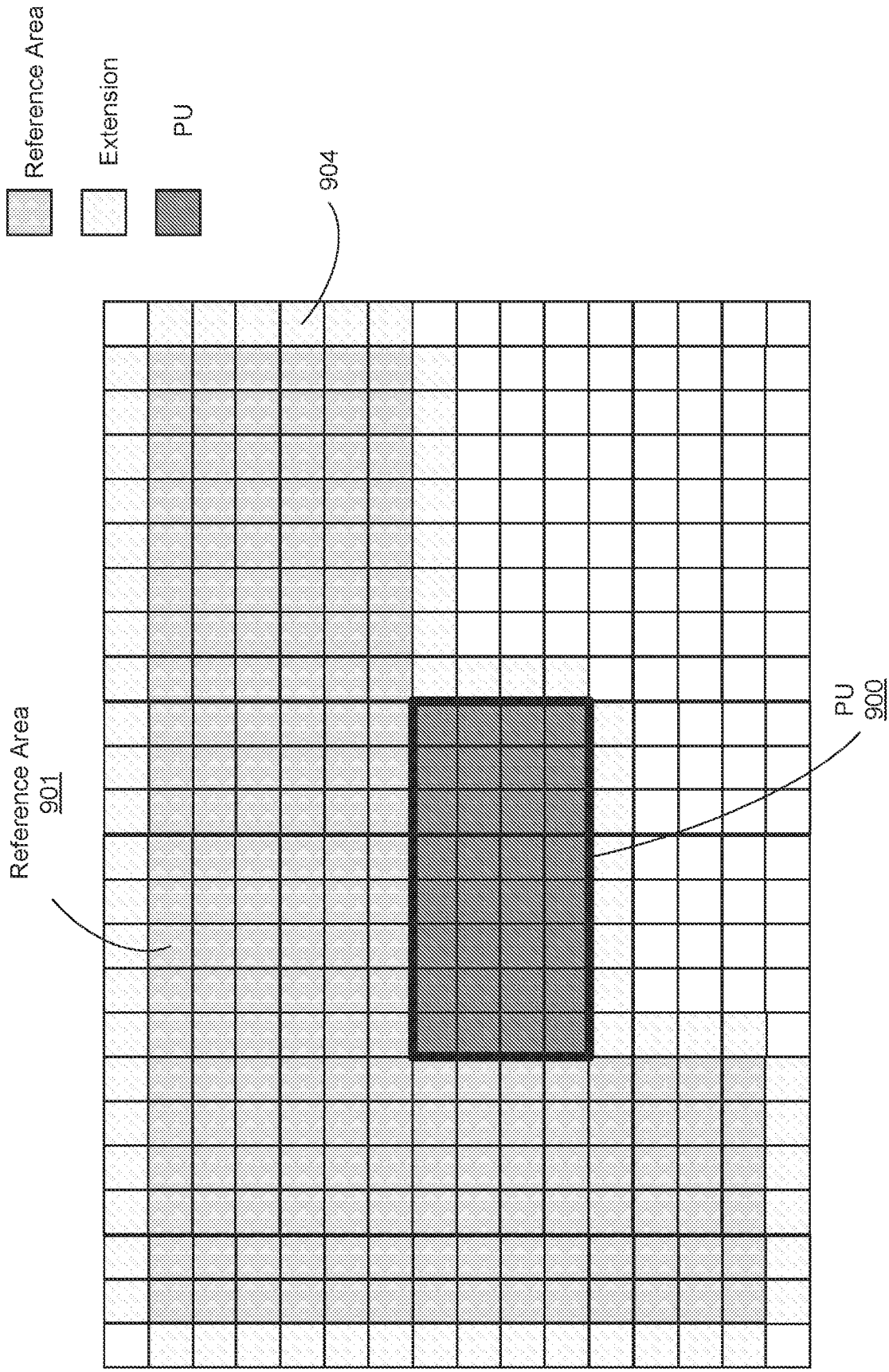


FIG. 9

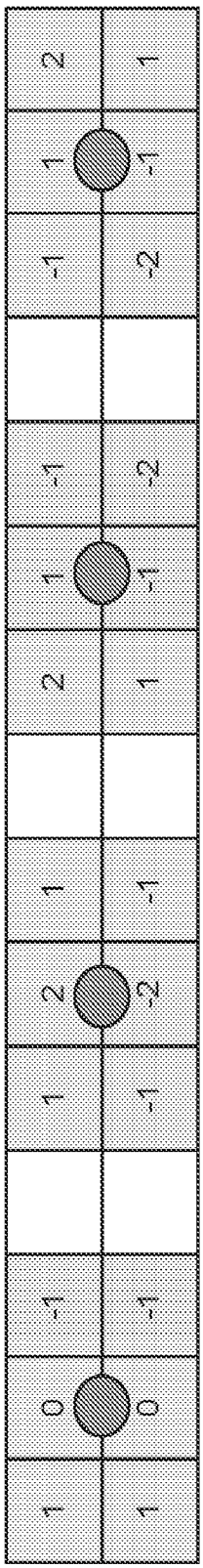


FIG. 10

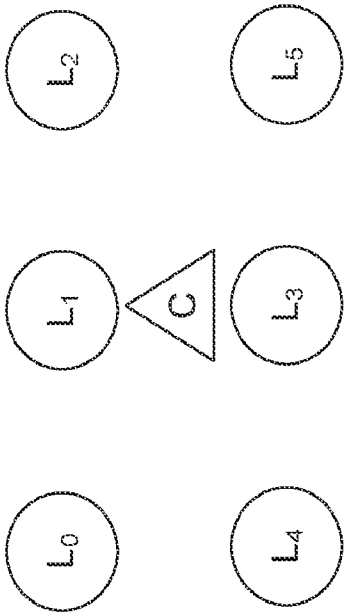


FIG. 11

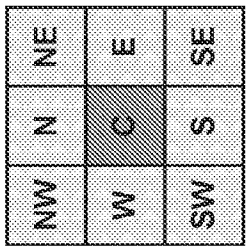


FIG. 12

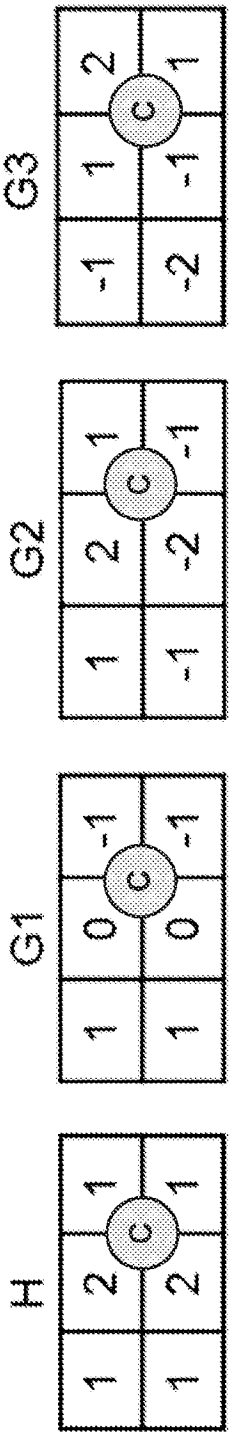


FIG. 13

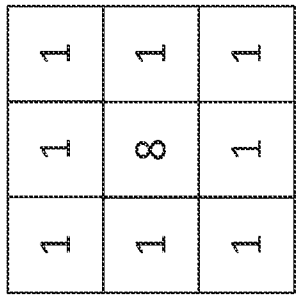


FIG. 14

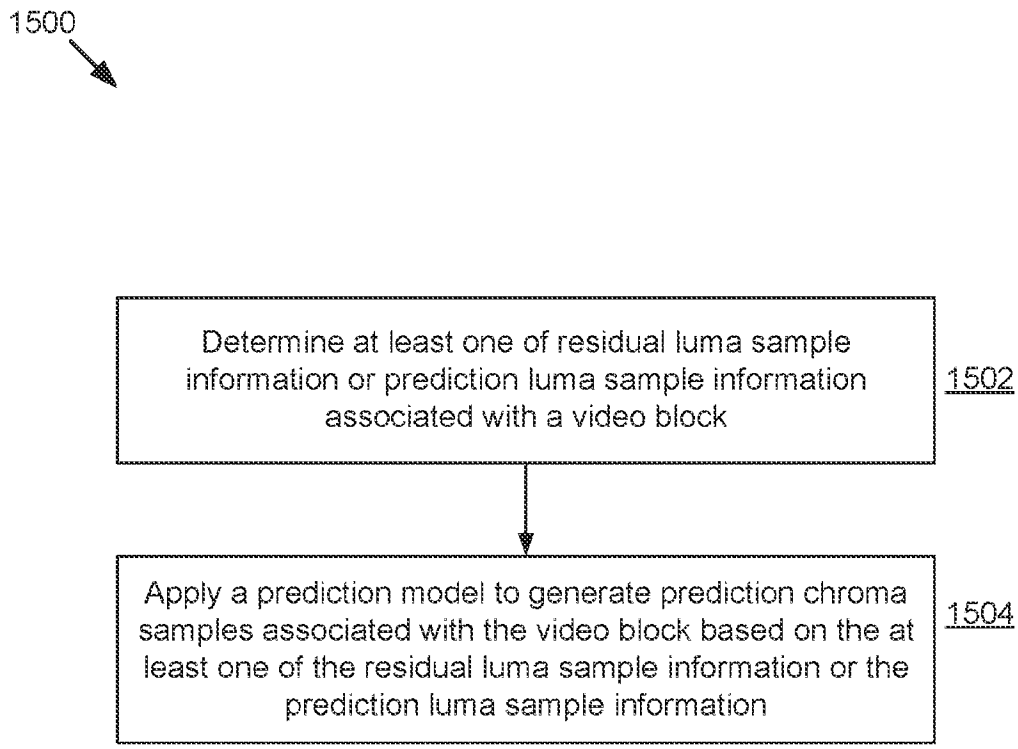

1500
Determine at least one of residual luma sample information or prediction luma sample information associated with a video block
1502
Apply a prediction model to generate prediction chroma samples associated with the video block based on the at least one of the residual luma sample information or the prediction luma sample information
1504

FIG. 15

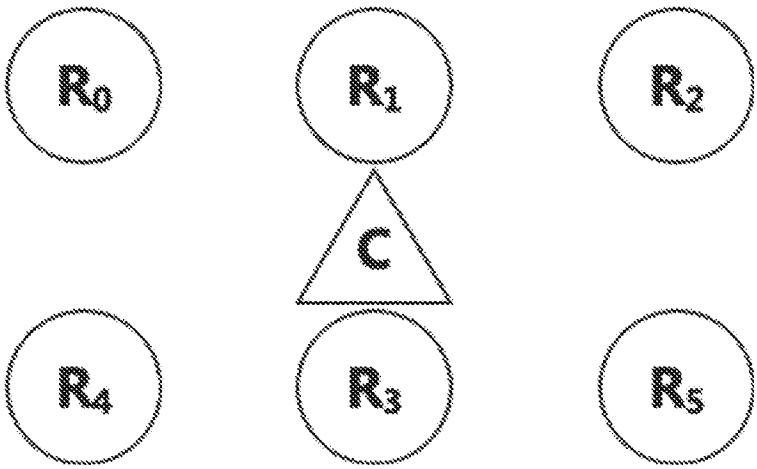


FIG. 16

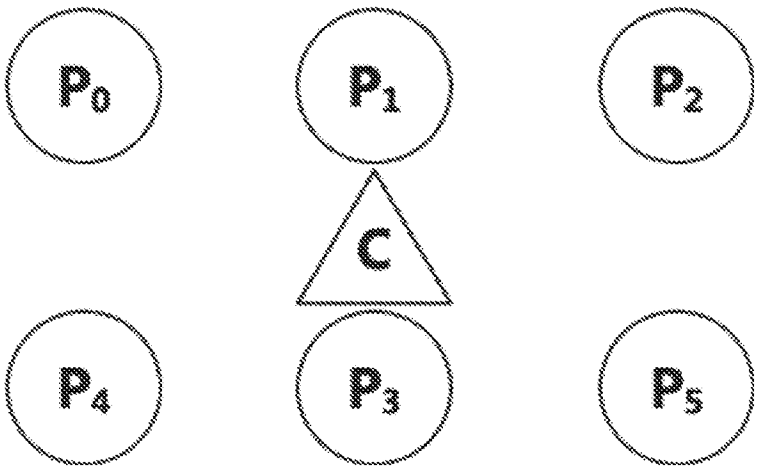


FIG. 17

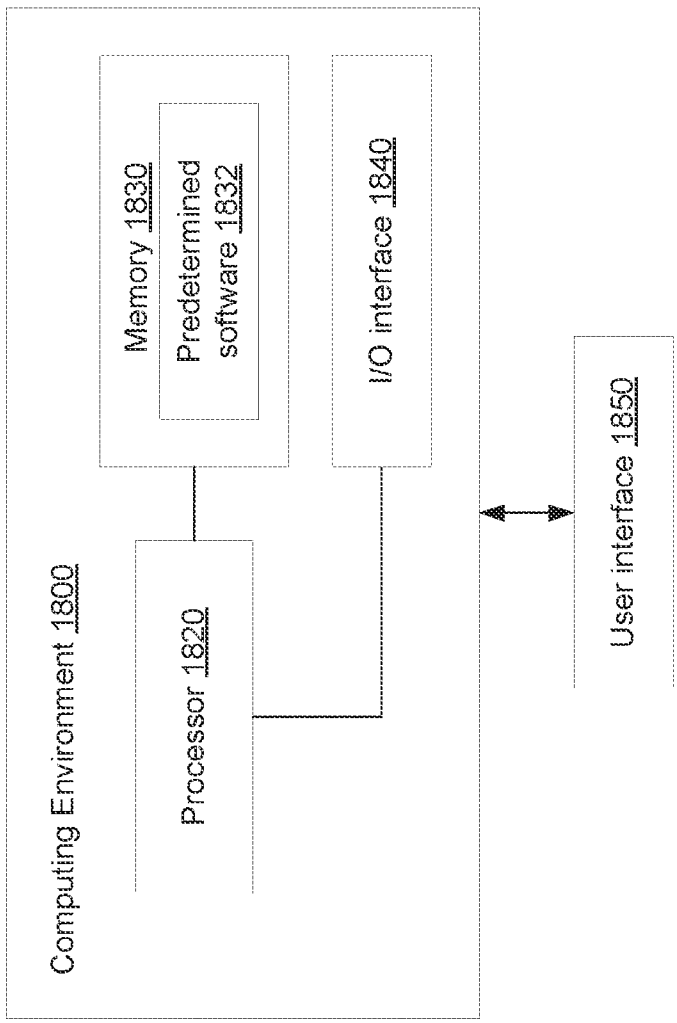

Computing Environment 1800
Processor 1820
Memory 1830
Predetermined software 1832
I/O interface 1840
User interface 1850

FIG. 18

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2024/036838

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/186(2014.01)i; **H04N 19/176**(2014.01)i; **H04N 19/132**(2014.01)i; **H04N 19/117**(2014.01)i; **H04N 19/149**(2014.01)i

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 19/186(2014.01); H04N 19/105(2014.01); H04N 19/132(2014.01); H04N 19/176(2014.01); H04N 19/196(2014.01); H04N 19/593(2014.01)

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

Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: prediction model, residual luma samples, prediction luma samples, reconstructed luma samples, prediction chroma samples

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2018-0077426 A1 (QUALCOMM INCORPORATED) 15 March 2018 (2018-03-15) paragraphs [0101], [0163]; and claims 1, 3	1-2,7,12,16-18,23,28,32-38
A		3-6,8-11,13-15,19-22,24-27,29-31
Y	US 2023-0050261 A1 (BEIJING DAJIA INTERNET INFORMATION TECHNOLOGY CO., LTD.) 16 February 2023 (2023-02-16) paragraphs [0064], [0187]; and claim 4	1-2,7,12,16-18,23,28,32-38
Y	US 2023-0062509 A1 (BRITISH BROADCASTING CORPORATION) 02 March 2023 (2023-03-02) claim 17	16,32

☑ 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
"D" document cited by the applicant in the international application
"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
22 October 2024	**22 October 2024**

Name and mailing address of the ISA/KR	Authorized officer
Korean Intellectual Property Office **189 Cheongsa-ro, Seo-gu, Daejeon** **35208, Republic of Korea**	**YANG, Jeong Rok**
Facsimile No. **+82-42-481-8578**	Telephone No. **+82-42-481-5709**

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2022-0217368 A1 (BEIJING DAJIA INTERNET INFORMATION TECHNOLOGY CO., LTD.) 07 July 2022 (2022-07-07) paragraph [0008]	1-38
A	US 2022-0060749 A1 (LG ELECTRONICS INC.) 24 February 2022 (2022-02-24) claim 1	1-38

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/US2024/036838

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
US 2018-0077426 A1	15 March 2018	CN 109716771 A	03 May 2019
		CN 109716771 B	28 June 2022
		EP 3513559 A1	24 July 2019
		EP 3513559 B1	04 August 2021
		JP 7044765 B2	30 March 2022
		KR 10-2019-0046852 A	07 May 2019
		KR 10-2534901 B1	19 May 2023
		US 10652575 B2	12 May 2020
		WO 2018-053293 A1	22 March 2018
US 2023-0050261 A1	16 February 2023	CN 115315948 A	08 November 2022
		EP 4128770 A1	08 February 2023
		EP 4128770 A4	10 April 2024
		JP 2024-038198 A	19 March 2024
		JP 2024-129124 A	26 September 2024
		JP 7460792 B2	02 April 2024
		KR 10-2024-0112991 A	19 July 2024
		WO 2021-195569 A1	30 September 2021
US 2023-0062509 A1	02 March 2023	CN 115349256 A	15 November 2022
		EP 4101170 A1	14 December 2022
		KR 10-2022-0129091 A	22 September 2022
		WO 2021-156587 A1	12 August 2021
US 2022-0217368 A1	07 July 2022	CN 116847098 A	03 October 2023
		CN 116847098 B	26 March 2024
		EP 4228263 A1	16 August 2023
		JP 2023-078281 A	06 June 2023
		JP 7416993 B2	17 January 2024
		KR 10-2022-0051426 A	26 April 2022
		US 11962789 B2	16 April 2024
		US 12063377 B2	13 August 2024
		US 2023-0239482 A1	27 July 2023
		US 2023-0336749 A1	19 October 2023
		WO 2020-076835 A1	16 April 2020
US 2022-0060749 A1	24 February 2022	CN 105635737 A	01 June 2016
		CN 105635737 B	15 March 2019
		EP 2387242 A2	16 November 2011
		EP 2387242 A3	04 March 2015
		KR 10-2021-0025713 A	09 March 2021
		KR 10-2223526 B1	04 March 2021
		KR 10-2268821 B1	23 June 2021
		US 11695954 B2	04 July 2023
		US 2023-0300371 A1	21 September 2023
		WO 2011-126348 A2	13 October 2011
		WO 2011-126348 A3	26 January 2012