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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 May 2025 (15.05.2025)

WIPO | PCT

(10) International Publication Number
WO 2025/101477 A1

(51) International Patent Classification:
H04N 19/117 (2014.01) *H04N 19/132* (2014.01)
H04N 19/82 (2014.01) *H04N 19/186* (2014.01)

(21) International Application Number:
PCT/US2024/054502

(22) International Filing Date:
05 November 2024 (05.11.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/547,541 06 November 2023 (06.11.2023) 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 US only)*: **MA, Changyue** [CN/US]; 8910 University Center Lane, Suite 400, San Diego, California 92122 (US).

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

(74) Agent: **TAN, Hao**; Arch & Lake LLP, 203 N. LaSalle Street, Suite 2100, Chicago, Illinois 60601 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, 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).

(54) Title: METHODS AND DEVICES FOR ADAPTIVE LOOP FILTER AND CROSS- COMPONENT ADAPTIVE LOOP FILTER

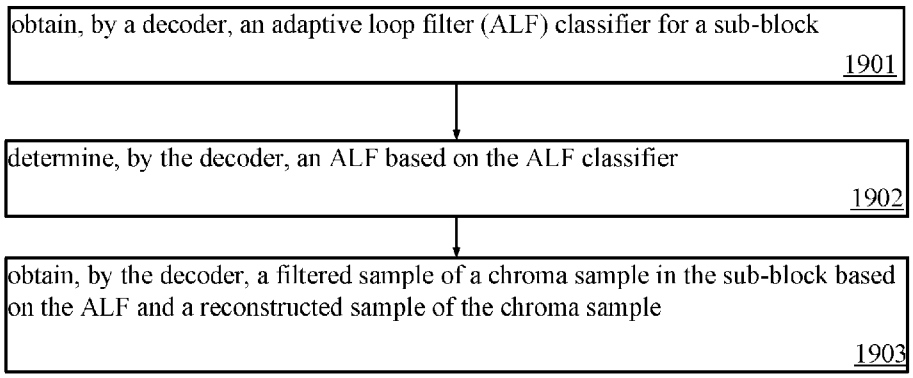


FIG. 19

(57) Abstract: Methods and apparatus are provided for video decoding/encoding. In a method provided, a decoder may obtain an adaptive loop filter (ALF) classifier for a sub-block, determine an ALF based on the ALF classifier, and obtain a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample.

WO 2025/101477 A1

[Continued on next page]

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

METHODS AND DEVICES FOR ADAPTIVE LOOP FILTER AND CROSS-COMPONENT ADAPTIVE LOOP FILTER

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application is based upon and claims priority to Provisional Application No. 63/547,541 filed on November 6, 2023, the content thereof is incorporated herein by reference in its entirety for all purposes.

TECHNICAL FIELD

[0002] The application is related to video coding and compression. More specifically, this application relates to methods and apparatus on improving the adaptive loop filtering process and cross-component adaptive loop filtering process.

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] Embodiments of the present disclosure provide for techniques relating to adaptive loop filtering and cross-component adaptive loop filtering.

[0005] In a first aspect, some embodiments of the present disclosure provide a method for video decoding. In the method, a decoder may obtain an adaptive loop filter (ALF) classifier for a sub-block. Further, the decoder may determine an ALF based on the ALF classifier. Then the decoder may obtain a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample.

[0006] In a second aspect, some embodiments of the present disclosure provide a method for video encoding. In the method, an encoder may obtain an adaptive loop filter (ALF) classifier for a sub-block. Further, the encoder may determine an ALF based on the ALF classifier. Then the encoder may obtain a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample.

[0007] In a third aspect of the present disclosure, some embodiments of the present disclosure provide an apparatus for video decoding. The apparatus may include one or more processors and a memory coupled to the one or more processors and configured to store instructions executable by the one or more processors. Furthermore, the one or more processors, upon execution of the instructions, are configured to perform the method according to the first aspect.

[0008] In a fourth aspect of the present disclosure, some embodiments of the present disclosure provide an apparatus for video encoding. The apparatus may include one or more processors and a memory coupled to the one or more processors and configured to store instructions executable by the one or more processors. Furthermore, the one or more processors, upon execution of the instructions, are configured to perform the method according to the second aspect.

[0009] In a fifth aspect of the present disclosure, some embodiments of the present disclosure provide a non-transitory computer readable storage medium for storing computer-executable instructions that, when executed by one or more computer processors, cause the one or more computer processors to perform the method according to the first aspect.

[0010] In a sixth aspect of the present disclosure, some embodiments of the present disclosure provide a non-transitory computer readable storage medium for storing computer-executable instructions that, when executed by one or more computer processors, cause the one or more computer processors to perform the method according to the second aspect.

[0011] In a seventh aspect of the present disclosure, some embodiments of the present disclosure provide a non-transitory computer-readable storage medium for storing a bitstream to be decoded by the method according to the first aspect.

[0012] In an eighth aspect of the present disclosure, some embodiments of the present disclosure provide a non-transitory computer-readable storage medium for storing a bitstream generated by the method according to the second aspect.

[0013] In a ninth aspect of the present disclosure, some embodiments of the present disclosure provide a method for receiving a bitstream, wherein the bitstream comprises encoded video information to be decoded by the method according to the first aspect.

[0014] In tenth aspect of the present disclosure, some embodiments of the present disclosure provide a method for transmitting a bitstream, wherein the bitstream comprises encoded video information generated by the method according to the second aspect.

[0015] 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

[0016] 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.

[0017] 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.

[0018] FIG. 2 is a block diagram illustrating an exemplary video encoder in accordance with some implementations of the present disclosure.

[0019] FIG. 3 is a block diagram illustrating an exemplary video decoder in accordance with some implementations of the present disclosure.

[0020] FIGS. 4A through 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.

[0021] FIG. 5 is a block diagram illustrating an exemplary video encoder in accordance with some embodiments of the present disclosure.

[0022] FIG. 6 is a block diagram illustrating an exemplary video decoder in accordance with some embodiments of the present disclosure.

[0023] FIG. 7 is an illustration of ALF filter shapes in accordance with some examples of the present disclosure.

[0024] FIG. 8 is a depiction of subsampled sample gradients in accordance with some examples of the present disclosure.

[0025] FIG. 9 is an illustration of a geometric transformation of a diamond filter shape in accordance with some examples of the present disclosure.

[0026] FIG. 10 is an illustration of an online filter shape used in an ECM in accordance with some examples of the present disclosure.

[0027] FIG. 11 is an illustration of a CCALF architecture, according to some examples of the present disclosure.

[0028] FIG. 12 is an illustration of a relative location of filtered chroma sample and its support in the luma plane for 4:2:0 chroma format with chroma location type 0.

[0029] FIG. 13 is an illustration of a 25-tap long filter according to some examples of the present disclosure.

[0030] FIG. 14 is an illustration of various online ALF filter inputs in accordance with some examples of the present disclosure.

[0031] FIG. 15 is an illustration of a filter shape for a prediction signal or before a SAO signal in accordance with examples of the present disclosure.

[0032] FIG. 16 is an illustration of an adjusted ALF filter shape in accordance with some examples of the present disclosure.

[0033] FIGS. 17A through 17C are diagrams illustrating symmetrical sample padding of luma ALF filtering in accordance with examples of the present disclosure.

[0034] FIG. 18 is a diagram illustrating a computing environment coupled with a user interface, according to some implementations of the present disclosure.

[0035] FIG. 19 is a flowchart illustrating a method for video decoding in accordance with some examples of the present disclosure.

[0036] FIG. 20 is a flowchart illustrating a method for video encoding in accordance with some examples of the present disclosure.

DETAILED DESCRIPTION

[0037] 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 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, the subject matter presented herein can be implemented on many types of electronic devices with digital video capabilities.

[0038] It should be illustrated 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.

[0039] 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 cloud servers, server computers, 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.

[0040] 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 an embodiment, 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.

[0041] 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.

[0042] 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.

[0043] 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. The encoded video data may comprise a sequence of pictures, each of which may comprise one or more sample arrays, for example, luma (Y) only for monochrome; luma and two chroma in YCbCr or YCgCo domain; or green, blue, and red in GBR (also known as RGB) domain. For convenience of notation and terminology in this application, in some embodiments, variables and terms associated with each set of three sample arrays may be referred to as luma and chroma, where the two chroma arrays may be referred to as Cb and Cr, regardless of the actual color representation method in use. The video data may be in a chroma format of 4:0:0, 4:2:0, 4:2:2, or 4:4:4, but the present application is not limited thereto.

[0044] 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.

[0045] 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.

[0046] 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.

[0047] 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.

[0048] 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.

[0049] 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.

[0050] 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 illustrated 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 and two chroma components (which may represent Y, Cb and Cr in YCbCr domain; Y, Cg and Co in YCgCo domain; or G, B and R in RGB domain for convenience of notation and terminology in this application as described above) according to any other of the luma component and the two chroma components to modify said any component based on the selected offset. Further, it should also be illustrated that a first component mentioned herein may be any of the luma component and the two chroma components, a second component mentioned herein may be any other of the luma component and the two chroma components, and a third component mentioned herein may be a remaining one of the luma component and the two chroma components. 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.

[0051] 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.

[0052] 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.

[0053] 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.

[0054] 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.

[0055] 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.

[0056] 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.

[0057] 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.

[0058] 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.

[0059] 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.

[0060] 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.

[0061] 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.

[0062] 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.

[0063] 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.

[0064] 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.

[0065] 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.

[0066] 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.

[0067] 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.

[0068] 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.

[0069] 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.

[0070] 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 in some embodiments, 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.

[0071] 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.

[0072] 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.

[0073] 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.

[0074] 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.

[0075] 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.

[0076] 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.

[0077] 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.

[0078] 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.

[0079] 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.

[0080] 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.

[0081] 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.

[0082] 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.

[0083] 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.

[0084] 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.

[0085] 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.

[0086] 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.

[0087] 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.

[0088] 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.

[0089] 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.

[0090] 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.

[0091] 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.

[0092] 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.

[0093] 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.

[0094] This disclosure is related to video coding and compression. More specifically, this disclosure relates to methods and apparatus on improving the coding efficiency of adaptive loop filter (ALF) and cross-component adaptive loop filter (CCALF).

[0095] Various video coding techniques may be used to compress video data. Video coding is performed according to one or more video coding standards. For example, nowadays, some well-known video coding standards include Versatile Video Coding (VVC), High Efficiency Video Coding (HEVC, also known as H.265 or MPEG-H Part2) and Advanced Video Coding (AVC, also known as H.264 or MPEG-4 Part 10), which are jointly developed by ISO/IEC MPEG and ITU-T VCEG. AOMedia Video 1 (AV1) was developed by Alliance for Open Media (AOM) as a successor to its preceding standard VP9. Audio Video Coding (AVS), which refers to digital audio and digital video compression standard, is another video compression standard series developed

by the Audio and Video Coding Standard Workgroup of China. Most of the existing video coding standards are built upon the famous hybrid video coding framework i.e., using block-based prediction methods (e.g., inter-prediction, intra-prediction) to reduce redundancy present in video images or sequences and using transform coding to compact the energy of the prediction errors. An important goal of video coding techniques is to compress video data into a form that uses a lower bit rate while avoiding or minimizing degradations to video quality.

[0096] The first generation AVS standard includes Chinese national standard "Information Technology, Advanced Audio Video Coding, Part 2: Video" (known as AVS1) and "Information Technology, Advanced Audio Video Coding Part 16: Radio Television Video" (known as AVS+). It can offer around 50% bit-rate saving at the same perceptual quality compared to MPEG-2 standard. The AVS1 standard video part was promulgated as the Chinese national standard in February 2006. The second generation AVS standard includes the series of Chinese national standard "Information Technology, Efficient Multimedia Coding" (knows as AVS2), which is mainly targeted at the transmission of extra HD TV programs. The coding efficiency of the AVS2 is double of that of the AVS+. In May 2016, the AVS2 was issued as the Chinese national standard. Meanwhile, the AVS2 standard video part was submitted by Institute of Electrical and Electronics Engineers (IEEE) as one international standard for applications. The AVS3 standard is one new generation video coding standard for UHD video application aiming at surpassing the coding efficiency of the latest international standard HEVC. In March 2019, at the 68-th AVS meeting, the AVS3-P2 baseline was finished, which provides approximately 30% bit-rate savings over the HEVC standard. Currently, there is one reference software, called high performance model (HPM), is maintained by the AVS group to demonstrate a reference implementation of the AVS3 standard.

[0097] Like the HEVC, the AVS3 standard is built upon the block-based hybrid video coding framework. FIG. 5 gives the block diagram of a generic block-based hybrid video encoding system. The input video signal is processed block by block (called coding units (CUs)). Different from the HEVC which partitions blocks only based on quad-trees, in the AVS3, one coding tree unit (CTU) is split into CUs to adapt to varying local characteristics based on quad/binary/extended-quad-tree. Additionally, the concept of multiple partition unit type in the HEVC is removed, i.e., the separation of CU, prediction unit (PU) and transform unit (TU) does not exist in the AVS3; instead, each CU is always used as the basic unit for both prediction and

transform without further partitions. In the tree partition structure of the AVS3, one CTU is firstly partitioned based on a quad-tree structure. Then, each quad-tree leaf node can be further partitioned based on a binary and extended-quad-tree structure. In FIG. 5, spatial prediction and/or temporal prediction may be performed. Spatial prediction (or "intra prediction") uses pixels from the samples of already coded neighboring blocks (which are called reference samples) in the same video picture/slice to predict the current video block. Spatial prediction reduces spatial redundancy inherent in the video signal. Temporal prediction (also referred to as "inter prediction" or "motion compensated prediction") uses reconstructed pixels from the already coded video pictures to predict the current video block. Temporal prediction reduces temporal redundancy inherent in the video signal. Temporal prediction signal for a given CU is usually signaled by one or more motion vectors (MVs) which indicate the amount and the direction of motion between the current CU and its temporal reference. Also, if multiple reference pictures are supported, one reference picture index is additionally sent, which is used to identify from which reference picture in the reference picture store the temporal prediction signal comes. After spatial and/or temporal prediction, the mode decision block in the encoder chooses the best prediction mode, for example based on the rate-distortion optimization method. The prediction block is then subtracted from the current video block; and the prediction residual is de-correlated using transform and then quantized. The quantized residual coefficients are inverse quantized and inverse transformed to form the reconstructed residual, which is then added back to the prediction block to form the reconstructed signal of the CU. Further in-loop filtering, such as deblocking filter, sample adaptive offset (SAO) and adaptive in-loop filter (ALF) may be applied on the reconstructed CU before it is put in the reference picture store and used as reference to code future video blocks. To form the output video bit-stream, coding mode (inter or intra), prediction mode information, motion information, and quantized residual coefficients are all sent to the entropy coding unit to be further compressed and packed.

[0098] The first version of the HEVC standard was finalized in October 2013, which offers approximately 50% bit-rate saving or equivalent perceptual quality compared to the prior generation video coding standard H.264/MPEG AVC. Although the HEVC standard provides significant coding improvements than its predecessor, there is evidence that superior coding efficiency can be achieved with additional coding tools over HEVC. Based on that, both VCEG and MPEG started the exploration work of new coding technologies for future video coding

standardization. One Joint Video Exploration Team (JVET) was formed in Oct. 2015 by ITU-T VCEG and ISO/IEC MPEG to begin significant study of advanced technologies that could enable substantial enhancement of coding efficiency. One reference software called joint exploration model (JEM) was maintained by the JVET by integrating several additional coding tools on top of the HEVC test model (HM).

[0099] In Oct. 2017, the joint call for proposals (CfP) on video compression with capability beyond HEVC was issued by ITU-T and ISO/IEC. In Apr. 2018, 23 CfP responses were received and evaluated at the 10-th JVET meeting, which demonstrated compression efficiency gain over the HEVC around 40%. Based on such evaluation results, the JVET launched a new project to develop the new generation video coding standard that is named as Versatile Video Coding (VVC). In the same month, one reference software codebase, called VVC test model (VTM), was established for demonstrating a reference implementation of the VVC standard.

[00100] Like HEVC, the VVC is built upon the block-based hybrid video coding framework. FIG. 5 gives the block diagram of a generic block-based hybrid video encoding system. The input video signal is processed block by block (called coding units (CUs)). In VTM-1.0, a CU can be up to 128x128 pixels. However, different from the HEVC which partitions blocks only based on quad-trees, in the VVC, one coding tree unit (CTU) is split into CUs to adapt to varying local characteristics based on quad/binary/ternary-tree. Additionally, the concept of multiple partition unit type in the HEVC is removed, i.e., the separation of CU, prediction unit (PU) and transform unit (TU) does not exist in the VVC anymore; instead, each CU is always used as the basic unit for both prediction and transform without further partitions. In the multi-type tree structure, one CTU is firstly partitioned by a quad-tree structure. Then, each quad-tree leaf node can be further partitioned by a binary and ternary tree structure. As shown in FIG. 4E, there are five splitting types, quaternary partitioning, horizontal binary partitioning, vertical binary partitioning, horizontal ternary partitioning, and vertical ternary partitioning. In FIG. 5, spatial prediction and/or temporal prediction may be performed. Spatial prediction (or “intra prediction”) uses pixels from the samples of already coded neighboring blocks (which are called reference samples) in the same video picture/slice to predict the current video block. Spatial prediction reduces spatial redundancy inherent in the video signal. Temporal prediction (also referred to as “inter prediction” or “motion compensated prediction”) uses reconstructed pixels from the already coded video pictures to predict the current video block. Temporal prediction reduces temporal

redundancy inherent in the video signal. Temporal prediction signal for a given CU is usually signaled by one or more motion vectors (MVs) which indicate the amount and the direction of motion between the current CU and its temporal reference. Also, if multiple reference pictures are supported, one reference picture index is additionally sent, which is used to identify from which reference picture in the reference picture store the temporal prediction signal comes. After spatial and/or temporal prediction, the mode decision block in the encoder chooses the best prediction mode, for example based on the rate-distortion optimization method. The prediction block is then subtracted from the current video block; and the prediction residual is de-correlated using transform and quantized. The quantized residual coefficients are inverse quantized and inverse transformed to form the reconstructed residual, which is then added back to the prediction block to form the reconstructed signal of the CU. Further in-loop filtering, such as deblocking filter, sample adaptive offset (SAO) and adaptive in-loop filter (ALF) may be applied on the reconstructed CU before it is put in the reference picture store and used to code future video blocks. To form the output video bit-stream, coding mode (inter or intra), prediction mode information, motion information, and quantized residual coefficients are all sent to the entropy coding unit to be further compressed and packed to form the bit-stream.

[00101] FIG. 6 gives a general block diagram of a block-based video decoder. The video bit-stream is first entropy decoded at entropy decoding unit. The coding mode and prediction information are sent to either the spatial prediction unit (if intra coded) or the temporal prediction unit (if inter coded) to form the prediction block. The residual transform coefficients are sent to inverse quantization unit and inverse transform unit to reconstruct the residual block. The prediction block and the residual block are then added together. The reconstructed block may further go through in-loop filtering before it is stored in reference picture store. The reconstructed video in reference picture store is then sent out to drive a display device, as well as used to predict future video blocks.

[00102] The main focus of the disclosure is to improve the adaptive loop filter (ALF) and cross-component adaptive loop filter (CCALF). The related knowledge is elaborated in the following sections.

[00103] ***ALF in VVC***

[00104] ***Filter Shapes, Linear Filtering and Adaptive Clipping***

[00105] In VVC, ALF is applied to the output samples of SAO. Two filter shapes, 7×7 diamond shape and 5×5 diamond shape are supported for luma and chroma components, respectively, as shown in FIG. 7. In FIG. 7, each square corresponds to a luma or a chroma sample and the center square corresponds to a current to-be-filtered sample. The filter coefficients use point-symmetry and each integer filter coefficient is represented with 7-bit fractional precision. In addition, the sum of coefficients of one filter is equal to 128, which is the fixed-point representation of 1.0 with 7-bit fractional precision:

[00106] $2\sum_{i=0}^{N-2} c_i + c_{N-1} = 128$ (1),

[00107] where the number of coefficients N is equal to 13 and 7 for 7×7 and 5×5 filter shape, respectively.

[00108] A filtered sample value $\tilde{R}(x,y)$ at coordinates (x, y) is derived by applying coefficient c_i to the reconstructed sample values R(x, y) as follows:

[00109] $\tilde{R}(x,y) = \left[\sum_{i=0}^{N-2} c_i\big(R(x+x_i, y+y_i) + R(x-x_i, y-y_i)\big) + c_{N-1}R(x,y) + 64\right] \gg 7$ (2),

[00110] where $(x+x_i, y+y_i)$ and $(x-x_i, y-y_i)$ are the coordinates of the reconstructed samples corresponding to i-th coefficient c_i. Due to the constraint in equation (1), equation (2) can be written as:

[00111] $\tilde{R}(x,y) = R(x,y) +$

[00112] $\{\left[\sum_{i=0}^{N-2} c_i \left(\big(R(x+x_i, y+y_i) - R(x,y)\big) + \big(R(x-x_i, y-y_i) - R(x,y)\big)\right) + 64\right] \gg 7\}$ (3).

[00113] In VVC, the possibility to clip the difference between the neighboring sample value and the current to-be-filtered sample is added to equation (3) as follows:

[00114] $\tilde{R}(x,y) = R(x,y) + \{[\sum_{i=0}^{N-2} c_i f_i + 64] \gg 7\}$ (4),

[00115] where

[00116] $f_i = \min\big(b_i, max\big(-b_i, R(x+x_i, y+y_i) - R(x,y)\big)\big) +$

[00117] $\min\left(b_i, max\left(-b_i, R(x-x_i, y-y_i) - R(x,y)\right)\right)$ (5)

[00118] b_i is the clipping parameter for a coefficient c_i determined by a clipping index d_i. b_i is derived as follows:

[00119] $b_i = \begin{cases} 2^{BD}, when\ d_i = 0 \\ 2^{BD-1-2d_i}, otherwise \end{cases}$ (6),

[00120] where BD is the sample bit depth and d_i can be 0, 1, 2 or 3.

[00121] ***Luma Sub-Block Level Filter Adaptation***

[00122] In VVC, sub-block level filter adaption is only applied to luma component. Each 4×4 luma block is classified based on its directionality and 2D Laplacian activity. First, the values of sample gradients for horizontal, vertical and two diagonal directions are calculated:

[00123] $H_{k,l} = |2R(k,l) - R(k-1,l) - R(k+1,l)|,$

[00124] $V_{k,l} = |2R(k,l) - R(k,l-1) - R(k,l+1)|,$

[00125] $D0_{k,l} = |2R(k,l) - R(k-1,l-1) - R(k+1,l+1)|,$

[00126] $D1_{k,l} = |2R(k,l) - R(k-1,l+1) - R(k+1,l-1)|.$ (7)

[00127] Based on the sample gradients, sub-block horizontal gradient, g_h, vertical gradient, g_v, and two diagonal gradients, g_{d0} and g_{d1}, are calculated as

[00128] $g_h = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} H_{k,l},$

[00129] $g_v = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} V_{k,l},$

[00130] $g_{d0} = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} D0_{k,l},$

[00131] $g_{d1} = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} D1_{k,l}$ (8)

[00132] Indices i and j refer to the coordinates of the upper left sample in the 4×4 luma block. As it can be seen from equation (8), the sum of sample gradients within a 10×10 luma window that covers the target 4×4 block is used for classifying that block. To reduce the complexity, only gradient of every second sample in a 10×10 window is calculated as illustrated in FIG. 8. The values of other sample gradients are set to 0.

[00133] Second, to assign the directionality D, the ratio of the maximum and the minimum of the sub-block horizontal and vertical gradients

[00134] $g_{h,v}^{max} = \max(g_h, g_v), g_{h,v}^{min} = \min(g_h, g_v),$ (9)

[00135] and the ratio of the maximum and the minimum of two sub-block diagonal gradients

[00136] $g_{d0,d1}^{max} = \max(g_{d0}, g_{d1}), g_{d0,d1}^{min} = \min(g_{d0}, g_{d1}),$ (10)

[00137] are compared against each other with a set of thresholds t_1 and t_2:

[00138] Step 1: If both $g_{h,v}^{max} \leq t_1 \cdot g_{h,v}^{min}$ and $g_{d0,d1}^{max} \leq t_1 \cdot g_{d0,d1}^{min}$, D is set to 0.

[00139] Step 2: If $g_{h,v}^{max}/g_{h,v}^{min} > g_{d0,d1}^{max}/g_{d0,d1}^{min}$, the directionality D is calculated in Step 3, otherwise in Step 4.

[00140] Step 3: If $g_{h,v}^{max} > t_2 \cdot g_{h,v}^{min}$, D is set to 2, otherwise D is set to 1.

[00141] Step 4: If $g_{d0,d1}^{max} > t_2 \cdot g_{d0,d1}^{min}$, D is set to 4, otherwise D is set to 3.

[00142] Each subsequent step in the above calculation of D is only executed if there is no value assigned to D in the previous steps. Third, an activity value A is calculated as

[00143] $\mathrm{A} = \left(\sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} (V_{k,l} + H_{k,l})\right) >> (BD - 2)$ (11)

[00144] A is further mapped to the range of 0 to 4: $\hat{A} = Q_{\min(A,15)}$,where $\{Q_n\} = \{0,1,2,2,2,2,2,3,3,3,3,3,3,3,3,4\}$. Finally, each 4×4 luma block is categorized into one of the 25 classes:

[00145] $\mathrm{C} = 5\mathrm{D} + \hat{A}$ (12)

[00146] Each class can have its own filter assigned.

[00147] Before filtering each 4×4 luma block, a geometric transformation, such as 90-degree rotation, diagonal or vertical flip, is applied to the filter coefficients, as illustrated in FIG. 9, depending on the sub-block gradient value as specified in Table 1.

[00148] **Table 1** Geometric transformation based on sub-block gradient values

Sub-block gradient values	Transformation
$g_{d1} < g_{d0}$ and $g_h < g_v$	No transformation
$g_{d1} < g_{d0}$ and $g_v \leq g_h$	Diagonal flip

$g_{do} \leq g_{d1}$ and $g_h < g_v$	Vertical flip
$g_{do} \leq g_{d1}$and $g_v \leq g_h$	90-degree rotation

[00149] ***Coding Tree Block Level Filter Adaptation***

[00150] In addition to the luma 4×4 block-level filter adaptation, ALF supports CTB-level filter adaptation. A luma CTB can use a filter set calculated for the current slice or one of the filter sets calculated for the already coded slices. It can also use one of the 16 offline trained filter sets. Within each luma CTB, which filter from the chosen filter set should be applied to each 4 × 4 block, is determined by the class C calculated in equation (12) for this block.

[00151] Chroma uses only CTB-level filter adaptation. Up to 8 filters can be used for chroma components in a slice. Each CTB can select one of these filters.

[00152] ***Syntax Design***

[00153] Filter coefficients and clipping indices are carried in ALF APSs. An ALF APS can include up to 8 chroma filters and one luma filter set with up to 25 filters. An index i_C is also included for each of the 25 luma classes. Classes having the same index i_C share the same filter. By merging different classes, the number of bits required to represent the filter coefficients is reduced. The absolute value of a filter coefficient is represented using a 0th order Exp-Golomb code followed by a sign bit for a non-zero coefficient. When clipping is enabled, a clipping index is also signaled for each filter coefficient using a two-bit fixed-length code. The storage needed for ALF coefficients and clipping indices within an APS is at most 3480 bits. Up to 8 ALF APSs can be used by the decoder at the same time.

[00154] Filter control syntax elements include two types of information. First, ALF on/off flags are signaled at sequence, picture, slice and CTB levels. Chroma ALF can be enabled at picture and slice level only if luma ALF is enabled at the corresponding level. Second, filter usage information is signaled at picture, slice and CTB level, if ALF is enabled at that level. Referenced ALF APSs IDs are coded at a slice level or at a picture level if all the slices within the picture use

the same APSs. Luma component can reference up to 7 ALF APSs and chroma components can reference 1 ALF APS. For a luma CTB, an index is signaled indicating which ALF APS or offline trained luma filter set is used. For a chroma CTB, the index indicates which filter in the referenced APS is used.

[00155] ***Line Buffer Reduction***

[00156] To reduce the storage requirement for ALF, VVC employs line buffer boundary processing. In VVC, line buffer boundaries are placed 4 luma samples and 2 chroma samples above horizontal CTU boundaries. When applying ALF to a sample on one side of a line buffer boundary, samples on the other side of the line buffer boundary cannot be used.

[00157] ***ALF in ECM***

[00158] ***ALF simplification removal***

[00159] ALF gradient subsampling and ALF virtual boundary processing are removed. Block size for classification is reduced from 4x4 to 2x2. Filter size for both luma and chroma, for which ALF coefficients are signalled, is increased to 9x9.

[00160] ***ALF with fixed filters***

[00161] To filter a luma sample, three different classifiers (C_0, C_1 and C_2) and three different sets of filters (F_0, F_1 and F_2) are used. Sets F_0 and F_1 contain fixed filters, with coefficients trained for classifiers C_0 and C_1. Coefficients of filters in F_2 are signalled. Which filter from a set F_i is used for a given sample is decided by a class C_i assigned to this sample using classifier C_i.

[00162] ***Filtering***

[00163] At first, two 13x13 diamond shape fixed filters F_0 and F_1 are applied to derive two intermediate samples $R_0(x,y)$ and $R_1(x,y)$. After that, F_2 is applied to $R_0(x,y)$, $R_1(x,y)$, neighboring samples, and samples before deblocking filter (DBF) to derive a filtered sample as

[00164] $\tilde{R}(x,y) = R(x,y) + \left[\sum_{i=0}^{19} c_i\left(f_{i,0} + f_{i,1}\right)\right] + \left[\sum_{i=20}^{21} c_i g_i\right] + \left[\sum_{i=22}^{24} c_i\left(h_{i,0} + h_{i,1}\right)\right]$ (13),

[00165] where $f_{i,j}$ is the clipped difference between a neighboring sample and current sample $R(x,y)$, g_i is the clipped difference between $R_{i-20}(x,y)$ and current sample $R(x,y)$, $h_{i,j}$ is the clipped difference between a neighboring sample before DBF and current sample $R(x,y)$. The filter coefficients c_i, i=0,…24, are signaled. The filter shape of F_2 is presented in FIG. 10.

[00166] ***Classification***

[00167] Based on directionality D_i and activity $\hat{A}_i$, a class C_i is assigned to each 2x2 block:

[00168] $C_i = \hat{A}_i * M_{D,i} + D_i$ (14)

[00169] where $M_{D,i}$ represents the total number of directionalities D_i.

[00170] As in VVC, values of the horizontal, vertical, and two diagonal gradients are calculated for each sample using 1-D Laplacian. The sum of the sample gradients within a 4×4 window that covers the target 2×2 block is used for classifier C_0 and the sum of sample gradients within a 12×12 window is used for classifiers C_1 and C_2. The sums of horizontal, vertical and two diagonal gradients are denoted, respectively, as g_h^i, g_v^i, g_{d1}^i and g_{d2}^i. The directionality D_i is determined by comparing

[00171] $r_{h,v}^i = \frac{\max(g_h^i, g_v^i)}{\min(g_h^i, g_v^i)}$, $r_{d1,d2}^i = \frac{\max(g_{d1}^i, g_{d2}^i)}{\min(g_{d1}^i, g_{d2}^i)}$ (15)

[00172] with a set of thresholds. The directionality D_2 is derived as in VVC using thresholds 2 and 4.5. For D_0 and D_1, horizontal/vertical edge strength E_{HV}^i and diagonal edge strength E_D^i are calculated first. Thresholds Th=[1.25, 1.5, 2, 3, 4.5, 8] are used. Edge strength E_{HV}^i is 0 if $r_{h,v}^i \leq$Th[0]; otherwise, E_{HV}^i is the maximum integer such that $r_{h,v}^i >$Th[E_{HV}^i-1]. Edge strength E_D^i is 0 if $r_{d1,d2}^i \leq$Th[0]; otherwise, E_D^i is the maximum integer such that $r_{d1,d2}^i >$Th[E_D^i-1]. When $r_{h,v}^i > r_{d1,d2}^i$, i.e., horizontal/vertical edges are dominant, the D_i is derived by using Table 2 (a); otherwise, diagonal edges are dominant, the D_i is derived by using Table 2 (b).

[00173] **Table 2.** Mapping of E_D^i and E_{HV}^i to D_i

(a)

E_{HV}^{i} \ E_{D}^{i}	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	1	2	0	0	0	0	0
2	3	4	5	0	0	0	0
3	6	7	8	9	0	0	0
4	10	11	12	13	14	0	0
5	15	16	17	18	19	20	0
6	21	22	23	24	25	26	27

(b)

E_{D}^{i} \ E_{HV}^{i}	0	1	2	3	4	5	6
0	28	0	0	0	0	0	0
1	29	30	0	0	0	0	0
2	31	32	33	0	0	0	0
3	34	35	36	37	0	0	0
4	38	39	40	41	42	0	0
5	43	44	45	46	47	48	0
6	49	50	51	52	53	54	55

[00174] To obtain $\hat{A}_i$, the sum of vertical and horizontal gradients A_i is mapped to the range of 0 to n, where n is equal to 4 for $\hat{A}_2$ and 15 for $\hat{A}_0$ and $\hat{A}_1$.

[00175] In an ALF_APS, up to 4 luma filter sets are signalled, each set may have up to 25 filters.

[00176] ***Alternative 2x2 ALF classifier***

[00177] Classification in ALF is extended with an additional alternative classifier. For a signalled luma filter set, a flag is signalled to indicate whether the alternative classifier is applied. Geometrical transformation is not applied to the alternative band classifier. When the band-based classifier is applied, the sum of sample values of a 2x2 luma block is calculated at first. Then the class index is calculated as below,

[00178] class_index = (sum * 25) >> (sample bit depth + 2) (16).

[00179] ***Residual based classifier***

[00180] Classification in ALF is extended with a third classifier based on luma residual sample values. For each 2x2 luma block, the sum of absolute values of the residual samples in a neighbouring 8x8 window is calculated, and the class index is derived as:

classIdx = sum >> (sample bit depth – 4).

[00181] The value of classIdx is in the range of 0 to 24, same as in ECM-8.0. The classifier usage is signalled for each luma filter set in APS.

[00182] ***CCALF in VVC***

[00183] ***Filter Shapes and Precision***

[00184] CCALF uses the luma sample values to refine the chroma sample values within the ALF process. As shown in FIG. 11, a linear filtering operation takes the luma sample values as input and generates the correction values for the chroma sample values. The correction is generated independently for each

[00185] chroma component i, $i \in \{\mathrm{Cb}, \mathrm{Cr}\}$ and can be represented by: $\Delta R_i(x,y) = \sum_{(x_0,y_0)\in S_i} R_Y(x_C + x_0, y_C + y_0) c_i(x_0, y_0)$, where (x,y) is the sample location of the chroma component i, (x_C, y_C) is the luma sample location derived from (x,y), (x_0, y_0) are the filter support offset around (x_C, y_C), S_i is the filter support region in luma for the chroma component i. The luma location (x_C, y_C) is determined based on the spatial scaling factor between the luma and chroma planes. The sample values in the luma support region are also inputs to the ALF luma stage and correspond to the output of the SAO stage.

[00186] As shown in FIG. 12, the CCALF filter has a diamond shape. As seen in FIG. 12, for a 4:2:0 video sequence, with chroma location type 0, i.e., when the chroma samples are horizontally co-sited with the even numbered columns of the luma samples and vertically interstitial between the rows of the luma samples, the center of the diamond is aligned with a chroma sample location.

[00187] CCALF coefficients have a greater degree of flexibility compared to regular ALF coefficients, since no symmetry constraints are enforced. However, two limitations are enforced:

[00188] To preserve DC neutrality, the sum of CCALF coefficient values is required to be zero. As a result, only seven of the eight CCALF coefficients need to be signalled in the bitstream, and the coefficient at location (x_C, y_C) is derived at the decoder.

[00189] The absolute value of CCALF coefficients is restricted to be either zero or an integer power of two, specifically {0, 1, 2, 4, 8, 16, 32, 64}. This enables implementations to use variable bit-shift operations in place of multiplications for CCALF, if desired.

[00190] ***Syntax Design***

[00191] The maximum number of filters per chroma component of a picture was four in the final design of VVC. A different set of CCALF coefficients can be selected for each CTU of a chroma component. As is the case for the regular ALF coefficients, CCALF coefficients are signalled within an ALF APS. Each ALF APS contains up to four CCALF filters for each chroma component. While CCALF can be enabled at a sequence level, it can only be enabled if ALF is also enabled for the sequence. Similarly, CCALF can be enabled at picture and slice level only if luma ALF is enabled at the corresponding level.

[00192] ***Line Buffer Reduction***

[00193] As described in section 3.1.5, the luma and the chroma line buffer boundaries are four and two samples, respectively, above the CTU boundary. For the 4:2:0 chroma format, this results in line buffer boundaries that are aligned for chroma and luma. However, for 4:2:2 and 4:4:4 chroma formats, the chroma and the luma line buffer boundaries are not aligned with each other. As a result of this misalignment, for 4:2:2 and 4:4:4 chroma formats, CC-ALF is not applied to the rows three and four samples above the CTU boundary.

[00194] ***CCALF in ECM***

[00195] The CCALF process uses a linear filter to filter luma sample values and generate a residual correction for the chroma samples. A 25-tap large filter is used in CCALF process, which is illustrated in FIG. 13. For a given slice, the encoder can collect the statistics of the slice, analyze them and can signal up to 16 filters through APS.

[00196] Although ALF and CCALF have been improved in ECM, there is room to further improve the performance.

[00197] First, online ALF filter in ECM takes spatial neighboring pixels, fixed ALF filter results and spatial neighboring pixels before deblocking filter as input. However, besides these information, other information such as spatial neighboring pixels in prediction signal, spatial neighboring pixels in residual signal, or spatial neighboring pixels before SAO can also be used as online ALF filter equation input, which may benefit the coding performance.

[00198] Second, edge based classifier and band based classifier are used adaptively for online ALF filter in ECM. However, these two classifiers may be further combined to provide other classifiers, which may benefit the coding performance.

[00199] Third, the filter shape for chroma ALF is diamond in ECM, while the filter shape for luma ALF is long cross shape, such non-unified design may not be optimal from standardization point of view.

[00200] Fourth, the edge based classifier and band based classifier in ECM only consider the pixel values after SAO. However, after the pixel values from the stages: 1) right before deblocking filter 2) prediction signal 3) residual signal 4) right before SAO are saved as online ALF filter equation input, these pixel values can also be utilized to design new classifiers, which may benefit the coding performance.

[00201] Fifth, the edge based classifier and band based classifier in ECM only consider luma pixel values after SAO. However, the chroma pixel values can also be utilized to design new classifier, which may benefit the coding performance.

[00202] Sixth, similar to the luma pixel values from the stages: 1) right before deblocking filter 2) prediction signal 3) residual signal 4) right before SAO are saved as additional online luma ALF filter equation input, the chroma pixel values from the stages: 1) before deblocking filter 2) prediction signal 3) residual signal 4) right before SAO can also be saved as additional online chroma ALF filter equation input, which may benefit the coding performance.

[00203] Seventh, similar to the luma pixel values from the stages: 1) right before deblocking filter 2) prediction signal 3) residual signal 4) right before SAO are saved as additional online luma

ALF filter equation input, the luma pixel values from the stages: 1) right before deblocking filter 2) prediction signal 3) residual signal 4) right before SAO can also be saved as additional CCALF filter equation input, which may benefit the coding performance.

[00204] Eighth, the classifiers design in ECM only considers the reconstruction pixel values. However, the coding mode information such as whether a coding block is coded with skip mode, whether the coding block is coded with intra, inter P or inter B mode can also be utilized to design classifier, which may benefit the coding performance.

[00205] Ninth, after online ALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., according to current line buffer settings in VVC, additional line buffers are needed to save 4 rows of corresponding luma samples and 2 rows of corresponding chroma samples above horizontal CTU boundaries, which increases the implementation complexity.

[00206] Tenth, after CCALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., according to current line buffer settings in VVC, additional line buffers are needed to save 4 rows of corresponding luma samples above horizontal CTU boundaries, which increases the implementation complexity.

[00207] Eleventh, after online ALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., sample padding is needed when the filter shape of the additional input with its central position aligned with the to be filtered sample crosses a boundary. In some embodiments, the boundary may be a borderline of an area that includes the to be filtered samples. For example, the boundary may be a virtual boundary (line buffer boundary) or picture (slice, tile) boundary.

[00208] Twelfth, two edge based classifiers with two different window sizes are utilized for training two sets of ALF fixed filters. However, besides edge based classifiers, other classifiers such as band based classifier, residual based classifier, etc. may also be utilized for training

corresponding sets of ALF fixed filters, and online ALF filter may take outputs of all these trained sets of ALF fixed filters as additional inputs, which may benefit the coding performance.

[00209] Thirteenth, ALF fixed filters are trained using spatial neighboring reconstructed pixels as input. However, besides spatial neighboring reconstructed pixels, other spatial neighboring pixels such as spatial neighboring pixels right before deblocking filter, spatial neighboring pixels in prediction signal, spatial neighboring pixels in residual signal, or spatial neighboring pixels right before SAO may also be used as ALF fixed filters input when training the ALF fixed filters, which may benefit the coding performance.

[00210] Fourteenth, sub-block level filter adaption is only applied in luma ALF. However, besides luma ALF, sub-block level filter adaption may also be extended to chroma ALF, which may benefit the coding performance.

[00211] Fifteenth, sub-block level filter adaption is only applied in luma ALF. However, besides luma ALF, sub-block level filter adaption may also be extended to CCALF, which may benefit the coding performance.

[00212] Embodiments of the present disclosure provide methods and apparatus on improving the coding efficiency of adaptive loop filter (ALF) and cross component adaptive loop filter (CCALF).

[00213] Online ALF filter takes spatial neighboring pixels in prediction signal, spatial neighboring pixels in residual signal, or spatial neighboring pixels before SAO as additional input.

[00214] The classifiers which combine the features of edge based classifier and band based classifier are used as additional classifier for online ALF filter.

[00215] The filter shape for chroma ALF is changed from diamond shape to long cross shape to unify with the filter shape for luma ALF.

[00216] The classifiers which utilize the pixel values from the stages: 1) right before deblocking filter 2) prediction signal 3) residual signal 4) right before SAO are used as additional classifier for online ALF filter.

[00217] The classifiers which utilize the chroma pixel values are used as additional classifier for online ALF filter.

[00218] Online chroma ALF filter takes spatial neighboring pixels in chroma prediction signal, spatial neighboring pixels in chroma residual signal, spatial neighboring pixels from the stage right before chroma SAO, or spatial neighboring pixels from the stage right before chroma deblocking as additional input.

[00219] CCALF filter takes spatial neighboring pixels in luma prediction signal, spatial neighboring pixels in luma residual signal, spatial neighboring pixels from the stage right before luma SAO, or spatial neighboring pixels from the stage right before luma deblocking as additional input.

[00220] The classifiers which utilize the coding mode information such as whether a coding block is coded with skip mode, whether the coding block is coded with intra, inter P or inter B mode are used as additional classifiers for online ALF filter.

[00221] When online ALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., according to current line buffer settings in VVC, 4 rows of corresponding luma samples and 2 rows of corresponding chroma samples above horizontal CTU boundaries are assumed to default values, which may save these line buffers.

[00222] When CCALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples, 4) samples right before SAO, etc., according to current line buffer settings in VVC, 4 rows of corresponding luma samples above horizontal CTU boundaries are assumed to default values, which may save these line buffers.

[00223] When online ALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., sample padding is conducted when the filter shape of the additional input with its central

position aligned with the to be filtered sample crosses the virtual boundary (line buffer boundary) or picture (slice, tile) boundary.

[00224] Band based classifier, residual based classifier, etc. are utilized for training additional sets of ALF fixed filters. Then, the outputs of these additional sets of ALF fixed filters together with the outputs of the original two sets of ALF fixed filters trained based on the two edge based classifiers are utilized as the online ALF filter inputs.

[00225] When training ALF fixed filters, the spatial neighboring reconstructed pixels together with the spatial neighboring pixels right before deblocking filter, spatial neighboring pixels in prediction signal, spatial neighboring pixels in residual signal, or spatial neighboring pixels right before SAO are used as ALF fixed filter inputs.

[00226] Sub-block level filter adaption is applied to chroma ALF, where edge based classifier, band based classifier or residual based classifier are utilized in chroma ALF.

[00227] Sub-block level filter adaption is applied to CCALF, where edge base classifier, band based classifier or residual based classifier are utilized in CCALF.

[00228] According to an embodiment, 4 luma samples and 2 chroma samples above horizontal CTU boundaries (current line buffer settings in VVC, which can be adjusted according to customized settings) in before deblocking, prediction, residual, or before SAO signals are filled with predefined values when online ALF filter takes samples in before deblocking, prediction, residual, or before SAO signals as additional input.

[00229] According to an embodiment, 4 luma samples above horizontal CTU boundaries (current line buffer settings in VVC, which can be adjusted according to customized settings) in before deblocking, prediction, residual, or before SAO signals are filled with predefined values when CCALF filter takes luma samples in before deblocking, prediction, residual, or before SAO signals as additional input.

[00230] According to an embodiment, when online ALF filter takes samples right before deblocking, prediction samples, residual samples, or samples right before SAO as additional input,

sample padding is conducted when the filter shape of the additional input with its central position aligned with the to be filtered sample crosses the virtual boundary (line buffer boundary) or picture (slice, tile) boundary.

[00231] According to an embodiment, band based classifier, residual based classifier, etc. are utilized to train additional sets of ALF fixed filters. And, the outputs of these additional sets of ALF fixed filters together with the outputs of the original two sets of ALF fixed filters trained based on the two edge based classifiers are utilized as the online ALF filter inputs.

[00232] According to an embodiment, the spatial neighboring reconstructed pixels together with the spatial neighboring pixels right before deblocking filter, spatial neighboring pixels in prediction signal, spatial neighboring pixels in residual signal, or spatial neighboring pixels right before SAO are used as ALF fixed filter inputs when training ALF fixed filters.

[00233] According to an embodiment, sub-block level filter adaption is applied to chroma ALF, where edge based classifier, band based classifier or residual based classifier are utilized in chroma ALF.

[00234] According to an embodiment, sub-block level filter adaption is applied to CCALF, where edge base classifier, band based classifier or residual based classifier are utilized in CCALF.

[00235] In some embodiments of the present disclosure, the disclosed methods may be applied independently or jointly.

[00236] ***Information in prediction, residual or before SAO used as additional ALF input***

[00237] According to the one or more embodiments of the disclosure, information in prediction, residual or before SAO are used as additional ALF equation input. Different methods may be used to achieve this goal.

[00238] FIG. 14 presents the online ALF filter inputs. Online ALF filter can take all or a subset of prediction samples, output samples which are obtained by feeding prediction samples into the offline trained fixed filters, residual samples, output samples which are obtained by feeding residual samples into the offline trained fixed filters, reconstructed samples right before

SAO, and output samples which are obtained by feeding reconstructed samples right before SAO into the offline trained fixed filters, as additional inputs.

[00239] In the first method, it is proposed to take the spatial neighboring pixels in prediction signal as additional ALF equation input. Various filter shapes may be used to extract the information in prediction signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in prediction signal. In an embodiment, the clipping differences between the surrounding pixels in prediction signal and current pixel are used as ALF equation input. In another example, the clipping differences between the surrounding pixels in prediction signal and the collocated pixel in prediction signal, the clipping difference between the collocated pixel in prediction signal and current pixel are used as ALF equation input.

[00240] Besides applying additional online ALF filter taps directly to prediction signal, additional online ALF filter taps may also be applied to the midterm results which are obtained by feeding prediction signal to fixed filters. Various fixed filters may be applied to filter prediction signal to obtain the midterm results, which may gather the prediction signal information in a large receptive field. For example, the two 13x13 diamond shape fixed filters utilized in ALF in ECM may be utilized to filter prediction signal to obtain the midterm results. When applying fixed filters to prediction signal, the block level classification results may directly utilize the block level classification results computed for right after SAO signal, or recomputed based on prediction signal. When applying fixed filters to prediction signal, one fixed filter trained based on one block level classifier may be utilized to obtain one midterm result, or two or more fixed filters trained based on two or more block level classifiers may be utilized to obtain two or more midterm results. In video coding standards, there are usually several groups fixed filters prepared, and one group fixed filter may be chosen from them by a rate distortion optimization (RDO) process. For example, in ECM, one group fixed filter (contains two 13x13 diamond shape fixed filters) is chosen from two groups by RDO process, and the group index is transmitted to decoder. When

applying fixed filters to prediction signal, the group index for prediction signal may be same to the group index for right after SAO signal, or different from the group index for right after SAO signal based on a predefined criterion (In ECM, there are two groups, so if the group index for right after SAO signal is 0, then the group index for prediction signal is 1; if the group index for right after SAO signal is 1, then the group index for prediction signal is 0), or decided for prediction signal by RDO process, where no group index for prediction signal is needed to transmitted to decoder in the first and second cases and the group index for prediction signal needed to transmitted to decoder in the third case.

[00241] When applying additional online filter taps to the midterm results which are obtained by feeding prediction signal to fixed filters, various filter shapes may be used to extract the information in the midterm results. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in the midterm results. In an embodiment, the clipping differences between the surrounding pixels in the midterm results and current pixel are used as ALF equation input. In another example, the clipping differences between the surrounding pixels in the midterm results and the collocated pixel in the midterm results, the clipping difference between the collocated pixel in the midterm results and current pixel are used as ALF equation input.

[00242] It should be noted that the additional online ALF filter taps may be applied to only prediction signal, or only the midterm results which are obtained by feeding prediction signal to fixed filters, or both prediction signal and the midterm results which are obtained by feeding prediction signal to fixed filters. For example, in AI (all intra) test, the additional online ALF filter taps are applied to only prediction signal; in RA (random access) test, the additional online ALF filter taps are applied to both prediction signal and the midterm results which are obtained by feeding prediction signal to fixed filters.

[00243] In the second method, it is proposed to take the spatial neighboring pixels in residual signal as additional ALF equation input. Various filter shapes may be used to extract the

information in residual signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in residual signal. In an embodiment, the clipping results of the collocated pixel in residual signal are used as ALF equation input.

[00244] Besides applying additional online ALF filter taps directly to residual signal, additional online ALF filter taps may also be applied to the midterm results which are obtained by feeding residual signal to fixed filters. Various fixed filters may be applied to filter residual signal to obtain the midterm results, which may gather the residual signal information in a large receptive field. For example, the two 13x13 diamond shape fixed filters utilized in ALF in ECM may be utilized to filter residual signal to obtain the midterm results. In one or more examples, considering that for prediction and before SAO signals, the ranges are just same to range after SAO signal, i.e. (0, 1024), which are positive, but for residual signals, the range may be positive or negative. Thus, when applying fixed filters to residual signal, the filtering results may be clipped to different range such as (-1024, 1024), (-512, 512), (-256, 256), (-128, 128), and so on. When applying fixed filters to residual signal, the block level classification results may directly utilize the block level classification results computed for right after SAO signal, or recomputed based on residual signal. When applying fixed filters to residual signal, one fixed filter trained based on one block level classifier may be utilized to obtain one midterm result, or two or more fixed filters trained based on two or more block level classifiers may be utilized to obtain two or more midterm results. When applying fixed filters to residual signal, the group index for residual signal may be same to the group index for right after SAO signal, or different from the group index for right after SAO signal based on a predefined criterion (In ECM, there are two groups, so if the group index for right after SAO signal is 0, then the group index for residual signal is 1; if the group index for right after SAO signal is 1, then the group index for residual signal is 0), or decided for residual signal by the RDO process, where no group index for residual signal is needed to transmitted to decoder in the first

and second cases and the group index for residual signal needed to transmitted to decoder in the third case.

[00245] When applying additional online filter taps to the midterm results which are obtained by feeding residual signal to fixed filters, various filter shapes may be used to extract the information in the midterm results. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in the midterm results. In an embodiment, the clipping results of the collocated pixel in the midterm results are used as ALF equation input.

[00246] It should be noted that the additional online ALF filter taps may be applied to only residual signal, or only the midterm results which are obtained by feeding residual signal to fixed filters, or both residual signal and the midterm results which are obtained by feeding residual signal to fixed filters. For example, in AI (all intra) test, the additional online ALF filter taps are applied to only residual signal; in RA (random access) test, the additional online ALF filter taps are applied to both residual signal and the midterm results which are obtained by feeding residual signal to fixed filters.

[00247] In the third method, it is proposed to take the spatial neighboring pixels from the stage right before SAO signal as additional ALF equation input. Various filter shapes may be used to extract the information in before SAO signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in before SAO signal. In an embodiment, the clipping differences between the surrounding pixels in before SAO signal and current pixel are used as ALF equation input. In another example, the clipping differences between the surrounding pixels in before SAO signal and the collocated pixel in before SAO signal, the clipping difference between the collocated pixel in before SAO signal and current pixel are used as ALF equation input.

[00248] Besides applying additional online ALF filter taps directly to right before SAO signal, additional online ALF filter taps may also be applied to the midterm results which are

obtained by feeding right before SAO signal to fixed filters. Various fixed filters may be applied to filter right before SAO signal to obtain the midterm results, which may gather the right before SAO signal information in a large receptive field. For example, the two 13x13 diamond shape fixed filters utilized in ALF in ECM may be utilized to filter right before SAO signal to obtain the midterm results. When applying fixed filters to right before SAO signal, the block level classification results may directly utilize the block level classification results computed for right after SAO signal, or recomputed based on right before SAO signal. When applying fixed filters to right before SAO signal, one fixed filter trained based on one block level classifier may be utilized to obtain one midterm result, or two or more fixed filters trained based on two or more block level classifiers may be utilized to obtain two or more midterm results. When applying fixed filters to right before SAO signal, the group index for right before SAO signal may be same to the group index for right after SAO signal, or different from the group index for right after SAO signal based on a predefined criterion (In ECM, there are two groups, so if the group index for right after SAO signal is 0, then the group index for right before SAO signal is 1; if the group index for right after SAO signal is 1, then the group index for right before SAO signal is 0), or decided for right before SAO signal by a RDO process, where no group index for right before SAO signal is needed to transmitted to decoder in the first and second cases and the group index for right before SAO signal needed to transmitted to decoder in the third case.

[00249] When applying additional online filter taps to the midterm results which are obtained by feeding right before SAO signal to fixed filters, various filter shapes may be used to extract the information in the midterm results. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in the midterm results. In an embodiment, the clipping differences between the surrounding pixels in the midterm results and current pixel are used as ALF equation input. In another example, the clipping differences between the surrounding pixels in the midterm results and the collocated pixel in the

midterm results, the clipping difference between the collocated pixel in the midterm results and current pixel are used as ALF equation input.

[00250] It should be noted that the additional online ALF filter taps may be applied to only right before SAO signal, or only the midterm results which are obtained by feeding right before SAO signal to fixed filters, or both right before SAO signal and the midterm results which are obtained by feeding right before SAO signal to fixed filters. For example, in AI (all intra) test, the additional online ALF filter taps are applied to only right before SAO signal; in RA (random access) test, the additional online ALF filter taps are applied to both right before SAO signal and the midterm results which are obtained by feeding right before SAO signal to fixed filters.

[00251] In the fourth method, it is proposed to take the information in prediction, residual or before SAO signal as ALF equation input. The utilization method proposed in the first, second and third method may be combined to achieve the fourth method.

[00252] ***New classifiers combined the features of edge based classifier and band based classifier***

[00253] According to the one or more embodiments of the disclosure, the features of edge based classifier and band based classifier are combined to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00254] In the first method, it is proposed to first compute the directionality D of the sub-block of luma component, then the sum of sample values of the sub-block is calculated and it is mapped to the index referring to the band based classifier, and the class index for the sub-block is calculated as

[00255] $C = B * M_D + D$ (17),

[00256] where B is the index calculated referring to the band based classifier, M_D represents the total number of directionalities D. In an embodiment, for the 2x2 luma block, the directionality D is calculated the same to D_2 in ECM, and B is calculated as

[00257] B = (sum * 5) >> (sample bit depth + 2) (18).

[00258] In the second method, it is proposed to first compute the activity value A of the sub-block of luma component, then the sum of sample values of the sub-block is calculated and it is mapped to the index referring to the band based classifier, and the class index for the sub-block is calculated as

[00259] $C = B * M_A + A$ (19),

[00260] where B is the index calculated referring to the band based classifier, M_A represents the total number of the activity value A. In an embodiment, for the 2x2 luma block, the activity value A is calculated the same to $\hat{A}_2$ in ECM, and B is calculated as

[00261] B = (sum * 5) >> (sample bit depth + 2) (20).

[00262] In the third method, it is proposed to first compute the index of the sub-block of luma component referring to the edge based classifier, then the sum of sample values of the sub-block is calculated and it is mapped to the index referring to the band based classifier, and the class index for the sub-block is calculated as

[00263] $C = B * M_E + E$ (21),

[00264] where B is the index calculated referring to the band based classifier, M_E represents the total number of the index calculated referring to the edge based classifier, E is the index calculated referring to the edge based classifier. In an embodiment, for the 2x2 luma block, the index E is calculated the same to C_2 in ECM, and B is calculated as

[00265] B = (sum * 2) >> (sample bit depth + 2) (22).

[00266] ***Adjust the chroma ALF filter shape to unify with luma ALF filter shape***

[00267] In the third aspect of this disclosure, it is proposed to change the chroma ALF filter shape from diamond shape to long cross shape as shown in FIG. 16, which is unified with the luma ALF filter shape.

[00268] ***New classifiers utilized the pixel values from the stage right before deblocking filter***

[00269] According to the one or more embodiments of the disclosure, the pixel values from the stage right before deblocking filter are utilized to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00270] In the first method, it is proposed to first compute the directionality D of the sub-block of luma component, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before deblocking filter of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00271] $C = Dif * M_D + D$ (23),

[00272] where Dif is the difference index, M_D represents the total number of directionalities D. In an embodiment, for the 2x2 luma block, the directionality D is calculated the same to D_2 in ECM, and Dif is calculated as

[00273] $Dif = sum_{Dif} > 0? 2: (sum_{Dif} < 0? 0: 1)$ (24)

[00274] , where sum_{Dif} is the sum of difference values of the 2x2 luma block, or the sum of difference values in a neighboring NxN (such as 8x8) window which surrounds the 2x2 luma block.

[00275] In the second method, it is proposed to first compute the activity value A of the sub-block of luma component, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before deblocking filter of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00276] $C = Dif * M_A + A$ (25),

[00277] where Dif is the difference index, M_A represents the total number of the activity value A. In an embodiment, for the 2x2 luma block, the activity value A is calculated the same to $\hat{A}_2$ in ECM, and Dif is calculated as in equation (24).

[00278] In the third method, it is proposed to first compute the index of the sub-block of luma component referring to the edge based classifier, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before deblocking filter of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00279] $C = Dif * M_E + E$ (26),

[00280] where Dif is the difference index, M_E represents the total number of the index calculated referring to the edge based classifier, E is the index calculated referring to the edge based classifier. In an embodiment, for the 2x2 luma block, the index E is calculated the same to C_2 in ECM, and Dif is calculated as in equation (24).

[00281] In the fourth method, it is proposed to first compute the band index B of the sub-block of luma component, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before deblocking filter of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00282] $C = Dif * M_B + B$ (27),

[00283] where Dif is the difference index, M_B represents the total number of the band value. In an embodiment, for the 2x2 luma block, the band index B is calculated as

[00284] B = (sum * 8) >> (sample bit depth + 2) (28),

[00285] and Dif is calculated as in equation (24).

[00286] In the fifth method, it is proposed to compute the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before deblocking filter of the sub-block or in a neighboring NxN window which surrounds the sub-block, then the sum of difference values is mapped to the difference index and the difference index is used as the class index.

[00287] In the sixth method, it is proposed to calculate the edged based classifier or band based classifier based on the sample values from the stage right before deblocking filter, where the calculation method is same to original edge based classifier or band based classifier calculated based on the sample values after SAO.

[00288] ***New classifiers utilized the pixel values in prediction signal***

[00289] According to the one or more embodiments of the disclosure, the pixel values in prediction signal are utilized to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00290] In the first method, it is proposed to first compute the directionality D of the sub-block of luma component, then the sum of difference values between sample in after SAO and collocated sample in prediction signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00291] $C = Dif * M_D + D$ (29),

[00292] where Dif is the difference index, M_D represents the total number of directionalities D. In an embodiment, for the 2x2 luma block, the directionality D is calculated the same to D_2 in ECM, and Dif is calculated as

[00293] $Dif = sum_{Dif} > 0?\,2:(sum_{Dif} < 0?\,0:1)$ (30)

[00294] , where sum_{Dif} is the sum of difference values of the 2x2 luma block, or the sum of difference values in a neighboring NxN (such as 8x8) window which surrounds the 2x2 luma block.

[00295] In the second method, it is proposed to first compute the activity value A of the sub-block of luma component, then the sum of difference values between sample in after SAO and collocated sample in prediction signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00296] $C = Dif * M_A + A$ (31),

[00297] where Dif is the difference index, M_A represents the total number of the activity value A. In an embodiment, for the 2x2 luma block, the activity value A is calculated the same to $\hat{A}_2$ in ECM, and Dif is calculated as in equation (30).

[00298] In the third method, it is proposed to first compute the index of the sub-block of luma component referring to the edge based classifier, then the sum of difference values between sample in after SAO and collocated sample in prediction signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00299] $C = Dif * M_E + E$ (32),

[00300] where Dif is the difference index, M_E represents the total number of the index calculated referring to the edge based classifier, E is the index calculated referring to the edge based classifier. In an embodiment, for the 2x2 luma block, the index E is calculated the same to C_2 in ECM, and Dif is calculated as in equation (30).

[00301] In the fourth method, it is proposed to first compute the band index B of the sub-block of luma component, then the sum of difference values between sample in after SAO and collocated sample in prediction signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00302] $C = Dif * M_B + B$ (33),

[00303] where Dif is the difference index, M_B represents the total number of the band value. In an embodiment, for the 2x2 luma block, the band index B is calculated as

[00304] B = (sum * 8) >> (sample bit depth + 2) (34),

[00305] and Dif is calculated as in equation (30).

[00306] In the fifth method, it is proposed to compute the sum of difference values between sample in after SAO and collocated sample in prediction signal of the sub-block or in a neighboring

NxN window which surrounds the sub-block, then the sum of difference values is mapped to the difference index and the difference index is used as the class index.

[00307] In the sixth method, it is proposed to calculate the edged based classifier or band based classifier based on the sample values in prediction signal, where the calculation method is same to original edge based classifier or band based classifier calculated based on the sample values after SAO.

[00308] ***New classifiers utilized the pixel values in residual signal***

[00309] According to the one or more embodiments of the disclosure, the pixel values in residual signal are utilized to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00310] In the first method, it is proposed to first compute the directionality D of the sub-block of luma component, then the sum of pixel values in residual signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the residual index, and the class index for the sub-block is calculated as

[00311] $C = Resi * M_D + D$ (35),

[00312] where $Resi$ is the residual index, M_D represents the total number of directionalities D. In an embodiment, for the 2x2 luma block, the directionality D is calculated the same to D_2 in ECM, and $Resi$ is calculated as

[00313] $Resi = sum_{Resi} > 0?\,2:(sum_{Resi} < 0?\,0:1)$ (36)

[00314] , where sum_{Resi} is the sum of pixel values in residual signal of the 2x2 luma block, or the sum of pixel values in residual signal in a neighboring NxN (such as 8x8) window which surrounds the 2x2 luma block.

[00315] In the second method, it is proposed to first compute the activity value A of the sub-block of luma component, then the sum of pixel values in residual signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the residual index, and the class index for the sub-block is calculated as

[00316] $C = Resi * M_A + A$ (37),

[00317] where $Resi$ is the residual index, M_A represents the total number of the activity value A. In an embodiment, for the 2x2 luma block, the activity value A is calculated the same to $\hat{A}_2$ in ECM, and $Resi$ is calculated as in equation (36).

[00318] In the third method, it is proposed to first compute the index of the sub-block of luma component referring to the edge based classifier, then the sum of pixel values in residual signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the residual index, and the class index for the sub-block is calculated as

[00319] $C = Resi * M_E + E$ (38),

[00320] where $Resi$ is the residual index, M_E represents the total number of the index calculated referring to the edge based classifier, E is the index calculated referring to the edge based classifier. In an embodiment, for the 2x2 luma block, the index E is calculated the same to C_2 in ECM, and $Resi$ is calculated as in equation (36).

[00321] In the fourth method, it is proposed to first compute the band index B of the sub-block of luma component, then the sum of pixel values in residual signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the residual index, and the class index for the sub-block is calculated as

[00322] $C = Resi * M_B + B$ (39),

[00323] where $Resi$ is the residual index, M_B represents the total number of the band value. In an embodiment, for the 2x2 luma block, the band index B is calculated as

[00324] B = (sum * 8) >> (sample bit depth + 2) (40),

[00325] and $Resi$ is calculated as in equation (36).

[00326] In the fifth method, it is proposed to compute the sum of pixel values in residual signal of the sub-block or in a neighboring NxN window which surrounds the sub-block, then the

sum of residual values is mapped to the residual index and the residual index is used as the class index.

[00327] In the sixth method, it is proposed to first compute the sum of absolute value of the pixel values in residual signal of the sub block or in a neighboring NxN window which surrounds the sub-block and it is mapped to the absolute value of residual index $Resi_{Abso}$, then the sum of pixel values in residual signal of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the sign of residual index $Resi_{Sign}$, and the class index for the sub-block is calculated as

[00328] $C = Resi_{Sign} * M_{Abso} + Resi_{Abso}$ (41)

[00329] , where M_{Abso} represents the total number of the absolute value of residual index. In one example, for the 2x2 luma block, the absolute value of residual index $Resi_{Abso}$ is calculated as

[00330] $Resi_{Abso}$ = (Sum_{Abso} * 8) >> (sample bit depth + 2) (42)

[00331] , where Sum_{Abso} is the sum of absolute values of pixel values in residual signal of the 2x2 luma block, or the sum of absolute values of pixel values in residual signal in a neighboring NxN (such as 8x8) window which surrounds the 2x2 luma block, and $Resi_{Sign}$ is calculated as in equation (36). In this disclosure, N may be any integer based on application or other factors.

[00332] In the seventh method, it is proposed to compute the index of the sub-block based on the pixel values in residual signal referring to the edge based classifier, then the index is used as the class index.

[00333] In the eighth method, it is proposed to compute the index of the sub-block based on the absolute value of the pixel values in residual signal referring to the edge based classifier, then the index is used as the class index.

[00334] ***New classifiers utilized the pixel values from the stage right before SAO***

[00335] According to the one or more embodiments of the disclosure, the pixel values from the stage right before SAO are utilized to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00336] In the first method, it is proposed to first compute the directionality D of the sub-block of luma component, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before SAO of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00337] $C = Dif * M_D + D$ (43),

[00338] where Dif is the difference index, M_D represents the total number of directionalities D. In an embodiment, for the 2x2 luma block, the directionality D is calculated the same to D_2 in ECM, and Dif is calculated as

[00339] $Dif = sum_{Dif} > 0?\,2:(sum_{Dif} < 0?\,0:1)$ (44)

[00340] , where sum_{Dif} is the sum of difference values of the 2x2 luma block, or the sum of difference values in a neighboring NxN (such as 8x8) window which surrounds the 2x2 luma block. In this disclosure, N may be any integer based on application or other factors.

[00341] In the second method, it is proposed to first compute the activity value A of the sub-block of luma component, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before SAO of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00342] $C = Dif * M_A + A$ (45),

[00343] where Dif is the difference index, M_A represents the total number of the activity value A. In an embodiment, for the 2x2 luma block, the activity value A is calculated the same to $\hat{A}_2$ in ECM, and Dif is calculated as in equation (44).

[00344] In the third method, it is proposed to first compute the index of the sub-block of luma component referring to the edge based classifier, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before SAO of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00345] $C = Dif * M_E + E$ (46),

[00346] where Dif is the difference index, M_E represents the total number of the index calculated referring to the edge based classifier, E is the index calculated referring to the edge based classifier. In an embodiment, for the 2x2 luma block, the index E is calculated the same to C_2 in ECM, and Dif is calculated as in equation (44).

[00347] In the fourth method, it is proposed to first compute the band index B of the sub-block of luma component, then the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before SAO of the sub-block or in a neighboring NxN window which surrounds the sub-block is calculated and it is mapped to the difference index, and the class index for the sub-block is calculated as

[00348] $C = Dif * M_B + B$ (47),

[00349] where Dif is the difference index, M_B represents the total number of the band value. In an embodiment, for the 2x2 luma block, the band index B is calculated as

[00350] B = (sum * 8) >> (sample bit depth + 2) (48),

[00351] and Dif is calculated as in equation (44).

[00352] In the fifth method, it is proposed to compute the sum of difference values between sample from the stage right after SAO and collocated sample from the stage right before SAO of the sub-block or in a neighboring NxN window which surrounds the sub-block, then the sum of difference values is mapped to the difference index and the difference index is used as the class index. In this disclosure, N may be any integer based on application or other factors.

[00353] In the sixth method, it is proposed to calculate the edged based classifier or band based classifier based on the sample values from the stage right before SAO, where the calculation method is same to original edge based classifier or band based classifier calculated based on the sample values after SAO.

[00354] ***New classifiers utilized chroma pixel values***

[00355] According to the one or more embodiments of the disclosure, the chroma pixel values are utilized to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00356] In the first method, it is proposed to first compute the band index B_Y of the sub-block of luma component, then the band index B_U and B_V of the corresponding U and V components are computed, and the class index for the sub-block is calculated as

[00357] $C = B_Y * M_U * M_V + B_U * M_V + B_V$ (49),

[00358] where B_Y, B_U and B_V are the Y, U and V index calculated referring to the band based classifier, M_U and M_V represent the total number of the U and V band index value. In an embodiment, for the 2x2 luma block, the B_Y, B_U and B_V are calculated as

[00359] B_Y = (sumY * 6) >> (sample bit depth + 2) (50),

[00360] B_U = (sumU * 2) >> (sample bit depth + 2) (51),

[00361] B_V = (sumV * 2) >> (sample bit depth + 2) (52).

[00362] ***Chroma information from the stages right before deblocking, prediction, residual or right before SAO used as additional chroma ALF input***

[00363] According to the one or more embodiments of the disclosure, chroma information from the stages right before deblocking, prediction, residual or right before SAO are used as additional chroma ALF equation input. Different methods may be used to achieve this goal.

[00364] In the first method, it is proposed to take the spatial neighboring pixels in chroma prediction signal as additional chroma ALF equation input. Various filter shapes may be used to extract the information in chroma prediction signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in chroma prediction signal. In an embodiment, the clipping differences between the surrounding pixels in chroma prediction signal and current chroma pixel are used as chroma ALF equation input. In another example, the clipping differences between the surrounding pixels in chroma prediction signal and the collocated pixel in chroma prediction signal, the clipping difference between the collocated pixel in chroma prediction signal and current chroma pixel are used as chroma ALF equation input.

[00365] In the second method, it is proposed to take the spatial neighboring pixels in chroma residual signal as additional chroma ALF equation input. Various filter shapes may be used to extract the information in chroma residual signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in chroma residual signal. In an embodiment, the clipping results of the collocated pixel in chroma residual signal are used as chroma ALF equation input.

[00366] In the third method, it is proposed to take the spatial neighboring pixels from the stage right before chroma SAO signal as additional chroma ALF equation input. Various filter shapes may be used to extract the information from the stage right before chroma SAO signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information from the stage right before chroma SAO signal. In an embodiment, the clipping differences between the surrounding pixels from the stage right before chroma SAO signal and current chroma pixel are used as chroma ALF equation input. In another example, the clipping differences between the surrounding pixels from the stage right before chroma SAO signal and the collocated pixel from the stage right before chroma SAO signal, the

clipping difference between the collocated pixel from the stage right before chroma SAO signal and current chroma pixel are used as chroma ALF equation input.

[00367] In the fourth method, it is proposed to take the spatial neighboring pixels from the stage right before chroma deblocking signal as additional chroma ALF equation input. Various filter shapes may be used to extract the information from the stage right before chroma deblocking signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information from the stage right before chroma deblocking signal. In an embodiment, the clipping differences between the surrounding pixels from the stage right before chroma deblocking signal and current chroma pixel are used as chroma ALF equation input. In another example, the clipping differences between the surrounding pixels from the stage right before chroma deblocking signal and the collocated pixel from the stage right before chroma deblocking signal, the clipping difference between the collocated pixel from the stage right before chroma deblocking signal and current chroma pixel are used as chroma ALF equation input.

[00368] In the fifth method, it is proposed to take the information in chroma prediction, residual, before SAO or before deblocking signal as chroma ALF equation input. The utilization method proposed in the first, second third, fourth method may be combined to achieve the fifth method.

[00369] ***Luma information from the stages right before deblocking, prediction, residual or right before SAO used as additional CCALF input***

[00370] According to the one or more embodiments of the disclosure, luma information from the stages right before deblocking, prediction, residual or right before SAO are used as additional CCALF equation input. Different methods may be used to achieve this goal.

[00371] In the first method, it is proposed to take the spatial neighboring pixels in luma prediction signal as additional CCALF equation input. Various filter shapes may be used to extract the information in luma prediction signal. For example, the filter shape may be 3x4 as shown in FIG. 12. Various equation forms may be used to extract the information in luma prediction signal.

In an embodiment, the differences between the surrounding pixels in luma prediction signal and current corresponding luma pixel are used as CCALF equation input. In another example, the differences between the surrounding pixels in luma prediction signal and the collocated pixel in current corresponding luma prediction signal, the difference between the collocated pixel in current corresponding luma prediction signal and current corresponding luma pixel are used as CCALF equation input.

[00372] In the second method, it is proposed to take the spatial neighboring pixels in luma residual signal as additional CCALF equation input. Various filter shapes may be used to extract the information in luma residual signal. For example, the filter shape may be 3x4 as shown in FIG. 12. Various equation forms may be used to extract the information in luma residual signal. In an embodiment, the collocated pixel in luma residual signal are used as CCALF equation input.

[00373] In the third method, it is proposed to take the spatial neighboring pixels from the stage right before luma SAO signal as additional CCALF equation input. Various filter shapes may be used to extract the information from the stage right before luma SAO signal. For example, the filter shape may be 3x4 as shown in FIG. 12. Various equation forms may be used to extract the information from the stage right before luma SAO signal. In an embodiment, the differences between the surrounding pixels from the stage right before luma SAO signal and current corresponding luma pixel are used as CCALF equation input. In another example, the differences between the surrounding pixels from the stage right before luma SAO signal and the collocated pixel in current corresponding before luma SAO signal, the difference between the collocated pixel in current corresponding before luma SAO signal and current corresponding luma pixel are used as CCALF equation input.

[00374] In the fourth method, it is proposed to take the spatial neighboring pixels from the stage right before luma deblocking signal as additional CCALF equation input. Various filter shapes may be used to extract the information from the stage right before luma deblocking signal. For example, the filter shape may be 3x4 as shown in FIG. 12. Various equation forms may be

used to extract the information from the stage right before luma deblocking signal. In an embodiment, the differences between the surrounding pixels from the stage right before luma deblocking signal and current corresponding luma pixel are used as CCALF equation input. In another example, the differences between the surrounding pixels from the stage right before luma deblocking signal and the collocated pixel in current corresponding before luma deblocking signal, the difference between the collocated pixel in current corresponding before luma deblocking signal and current corresponding luma pixel are used as CCALF equation input.

[00375] In the fifth method, it is proposed to take the information in luma prediction, residual, before SAO or before deblocking signal as CCALF equation input. The utilization method proposed in the first, second third, fourth method may be combined to achieve the fifth method.

[00376] ***New classifiers utilized the coding mode information***

[00377] According to the one or more embodiments of the disclosure, the coding mode information such as whether the coding block is coded with skip mode, whether the coding block is coded with intra, inter P or inter B mode, is utilized to derive new classifiers for online ALF filter. Different methods may be used to achieve this goal.

[00378] In the first method, it is proposed to record whether the coding block is coded with skip mode during the encoding and decoding process, then this information is utilized to design a new classifier. In an embodiment, the classifier which has 2 classes corresponding to the skip mode is true or false is added as a new classifier. In another example, the classifier which combines the skip mode information with EO or BO is added as a new classifier.

[00379] In the second method, it is proposed to record whether the coding block is coded with intra mode, inter P mode, or inter B mode during the encoding and decoding process, then this information is utilized to design a new classifier. In an embodiment, the classifier which has 3 classes corresponding to the intra mode, inter P mode or inter B mode is added as a new classifier.

In another example, the classifier which combines the intra, inter P or inter B mode information with EO or BO is added as a new classifier.

[00380] In the third method, it is proposed to take both the coding mode information whether the coding block is coded with skip mode, whether the coding block is coded with intra, inter P or inter B mode to design the new classifier. The utilization method proposed in the first and second method may be combined to achieve the third method.

[00381] ***Line buffer reduction for additional ALF input***

[00382] According to the one or more embodiments of the disclosure, when online ALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., there would be line buffers to save these samples, to reduce the line buffer requirements for these additional inputs, according to current line buffer settings in VVC, 4 rows of corresponding luma samples and 2 rows of corresponding chroma samples above horizontal CTU boundaries are assumed to default values, which may save these line buffers. Different methods may be used to achieve this goal.

[00383] In the first method, according to current line buffer settings in VVC, it is proposed to assume 4 rows of luma residual samples and 2 rows of chroma residual samples above horizontal CTU boundaries to zero values, assume 4 rows of luma samples and 2 rows of chroma samples above horizontal CTU boundaries from the stages: 1) samples right before deblocking 2) prediction samples 3) samples right before SAO to collocated sample values from the stage samples right after SAO.

[00384] In the second method, according to current line buffer settings in VVC, it is proposed to assume 4 rows of luma samples and 2 rows of chroma samples above horizontal CTU boundaries from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc. in a repetitive manner with the corresponding nearest sample values in the horizontal CTU boundaries.

[00385] In the third method, according to current line buffer settings in VVC, it is proposed to assume 4 rows of luma samples and 2 rows of chroma samples above horizontal CTU boundaries from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc. in a mirrored manner, where the first row of luma samples and first row of chroma samples above horizontal CTU boundaries are assumed to the corresponding sample values in the horizontal CTU boundaries, the second row of luma samples and second row of chroma samples above horizontal CTU boundaries are assumed to the corresponding sample values in the first rows of samples below the horizontal CTU boundaries, and so on.

[00386] It should be noted that 4 rows of luma samples and 2 rows of chroma samples above horizontal CTU boundaries are current VVC line buffer settings, the specific values may be adjusted according to customized settings.

[00387] ***Line buffer reduction for additional CCALF input***

[00388] According to the one or more embodiments of the disclosure, when CCALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., there would be line buffers to save these samples, to reduce the line buffer requirements for these additional inputs, according to current line buffer settings in VVC, 4 rows of corresponding luma samples above horizontal CTU boundaries are assumed to default values, which may save these line buffers. Different methods may be used to achieve this goal.

[00389] In the first method, according to current line buffer settings in VVC, it is proposed to assume 4 rows of luma residual samples above horizontal CTU boundaries to zero values, assume 4 rows of luma samples above horizontal CTU boundaries from the stages: 1) samples right before deblocking 2) prediction samples 3) samples right before SAO to collocated sample values from the stage samples right after SAO.

[00390] In the second method, according to current line buffer settings in VVC, it is proposed to assume 4 rows of luma samples above horizontal CTU boundaries from the stages: 1)

samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc. in a repetitive manner with the corresponding nearest sample values in the horizontal CTU boundaries.

[00391] In the third method, according to current line buffer settings in VVC, it is proposed to assume 4 rows of luma samples above horizontal CTU boundaries from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc. in a mirrored manner, where the first row of luma samples above horizontal CTU boundaries are assumed to the corresponding sample values in the horizontal CTU boundaries, the second row of luma samples above horizontal CTU boundaries are assumed to the corresponding sample values in the first row of samples below the horizontal CTU boundaries, and so on.

[00392] It should be noted that 4 rows of luma samples above horizontal CTU boundaries are current VVC line buffer settings, the specific values may be adjusted according to customized settings.

[00393] ***Sample padding for additional ALF input***

[00394] According to the one or more embodiments of the disclosure, when online ALF filter takes samples as additional input from the stages: 1) samples right before deblocking 2) prediction samples 3) residual samples 4) samples right before SAO, etc., sample padding is conducted when the filter shape of the additional input with its central position aligned with the to be filtered sample crosses the virtual boundary (line buffer boundary) or picture (slice, tile) boundary. Different methods may be used to achieve this goal.

[00395] In the first method, symmetrical sample padding is applied when the filter shape of the additional input with its central position aligned with the to be filtered sample crosses the virtual boundary (line buffer boundary) or picture (slice, tile) boundary. For example, assume online ALF filter takes residual samples as additional input, the filter shape of the fixed filter to be applied to the residual signal or the filter shape of the online filter which directly applies to residual signal is 7x7, the filter shape of the residual signal with its central position aligned with the to be

filtered sample crosses the line buffer boundary, the symmetrical sample padding is conducted as shown in FIGS. 17A-C, where p_{12} masks the collocated residual pixel of the to be filtered sample, p_0 to p_{24} are the original residual samples, $p_0^{'}$ to $p_{24}^{'}$ are the modified residual sample values, Bold lines are line buffer boundaries. Shaded samples represent padded residual samples. In a word, with symmetrical sample padding, the additional input samples which are not in the same boundary side with the collocated additional input sample of the to be filtered sample and the additional input symmetrical samples which are in the same boundary side with the collocated additional input sample of the to be filtered sample are both modified in a symmetry manner.

[00396] In the second method, repetitive sample padding is applied when the filter shape of the additional input with its central position aligned with the to be filtered sample crosses the virtual boundary (line buffer boundary) or picture (slice, tile) boundary. With repetitive padding, the additional input samples which are not in the same boundary side with the collocated additional input sample of the to be filtered sample are padded in the same manner with the symmetrical sample padding, the additional input samples which are in the same boundary side with the collocated additional input sample of the to be filtered sample remain unchanged.

[00397] ***ALF fixed filters with additional classifiers***

[00398] According to the one or more embodiments of the disclosure, the band classifier, residual based classifier, etc. are utilized to train additional sets of ALF fixed filters. Then, the outputs of these additional sets of ALF fixed filters are utilized as additional online ALF filter inputs. Different methods may be used to achieve this goal.

[00399] In the first method, different band classifiers are first utilized to train different sets of ALF fixed filters. Different band classifiers may be defined based on different window sizes. For example, there are two band classifiers. For the first band classifier, the sum of sample values of a 2x2 luma block is calculated and mapped to the band classifier index as follows:

[00400] class_index = (sum * 25) >> (sample bit depth + 2)

[00401] . For the second band classifier, the sum of sample values in a neighboring 8x8 window which surrounds the 2x2 luma block is calculated and mapped to the band classifier index as follows:

[00402] class_index = (sum * 25) >> (sample bit depth + 6)

[00403] . Different band classifiers may also be defined based on different class numbers. For example, there are two band classifiers. For the first band classifier, the sum of sample values of a 2x2 luma block is calculated and mapped to the band classifier index as follows:

[00404] class_index = (sum * 25) >> (sample bit depth + 2)

[00405] , where the class number of the band classifier is 25. For the second band classifier, the sum of sample values of a 2x2 luma block is calculated and mapped to the band classifier index as follows:

[00406] class_index = (sum * 100) >> (sample bit depth + 2)

[00407] , where the class number of the band classifier is 100. Different taps of ALF fixed filters may be utilized. For example, the fixed filters are 13x13 diamond shape. After training different sets of ALF fixed filters based on different band classifiers, the intermediate results may be obtained by feeding the reconstructed pixel values to the new trained ALF fixed filters. Then, online ALF filters may take the intermediate results as additional inputs. Various filter shapes may be used to extract the information in the intermediate results. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to extract the information in the intermediate results. For example, the clipping differences between the surrounding pixels in the intermediate results and current pixel are used as additional online ALF filter input.

[00408] In the second method, different residual based classifiers are first utilized to train different sets of ALF fixed filters. Different residual based classifiers may be defined based on different window sizes. For example, there are two residual based classifiers. For the first residual based classifier, the sum of absolute values of the residual samples in a neighbouring 8x8 window

which surrounds the 2x2 luma block is calculated and mapped to the residual based classifier index as follows:

[00409] classIdx = sum >> (sample bit depth – 4)

[00410] , where the value of classIdx is in the range of 0 to 24. For the second residual based classifier, the sum of absolute values of the residual samples in a neighbouring 12x12 window which surrounds the 2x2 luma block is calculated and mapped to the residual based classifier index as follows:

[00411] classIdx = sum >> (sample bit depth – 4)

[00412] , where the value of classIdx is in the range of 0 to 24. Different residual based classifiers may also be defined based on different class numbers. For example, there are two residual based classifiers. For the first residual based classifier, the sum of absolute values of the residual samples in a neighbouring 8x8 window which surrounds the 2x2 luma block is calculated and mapped to the residual based classifier index as follows:

[00413] classIdx = sum >> (sample bit depth – 4)

[00414] , where the value of classIdx is in the range of 0 to 24. For the second residual based classifier, the sum of absolute values of the residual samples in a neighbouring 8x8 window which surrounds the 2x2 luma block is calculated and mapped to the residual based classifier index as follows:

[00415] classIdx = sum >> (sample bit depth – 4)

[00416] , where the value of classIdx is in the range of 0 to 49. Different taps of ALF fixed filters may be utilized. For example, the ALF fixed filters are 13x13 diamond shape. After training different sets of ALF fixed filters based on different residual based classifiers, the intermediate results may be obtained by feeding the reconstructed pixel values to the new trained ALF fixed filters. Then, online ALF filters may take the intermediate results as additional inputs. Various filter shapes may be used to extract the information in the intermediate results. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15. Various equation forms may be used to

extract the information in the intermediate results. For example, the clipping differences between the surrounding pixels in the intermediate results and current pixel are used as additional online ALF filter input.

[00417] In the third method, the methods presented in the first and second method may be combined. For example, one band classifier and one residual based classifier are utilized to train two sets of ALF fixed filters. Then, the outputs of the two new trained ALF fixed filters are utilized as additional inputs of the online ALF filters. It should be noted that besides the new trained ALF fixed filters, two ALF fixed filters which are trained based on edge based classifiers are already contained in original design of the ALF in ECM.

[00418] ***ALF fixed filters with additional inputs***

[00419] According to the one or more embodiments of the disclosure, the spatial neighboring pixels right before deblocking filter, spatial neighboring pixels in prediction signal, spatial neighboring pixels in residual signal, or spatial neighboring pixels right before SAO are used as additional ALF fixed filter inputs when training ALF fixed filters. Different methods may be used to achieve this goal.

[00420] In the first method, the spatial neighboring pixels right before deblocking filter are used as additional ALF fixed filter inputs when training ALF fixed filters. Various filter shapes may be used to extract the information in the spatial neighboring pixels right before deblocking filter. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15, or 13x13 diamond shape. Various equation forms may be used to extract the information in the spatial neighboring pixels right before deblocking filter. For example, the clipping differences between the surrounding pixels right before deblocking filter and current pixel are used as additional ALF fixed filter input.

[00421] In the second method, the spatial neighboring pixels in prediction signal are used as additional ALF fixed filter inputs when training ALF fixed filters. Various filter shapes may be used to extract the information in the prediction signal. For example, the filter shape may be 1×1,

3×3 or 5×5 as shown in FIG. 15, or 13x13 diamond shape. Various equation forms may be used to extract the information in the prediction signal. For example, the clipping differences between the surrounding pixels in the prediction signal and current pixel are used as additional ALF fixed filter input.

[00422] In the third method, the spatial neighboring pixels in residual signal are used as additional ALF fixed filter inputs when training ALF fixed filters. Various filter shapes may be used to extract the information in the residual signal. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15, or 13x13 diamond shape. Various equation forms may be used to extract the information in the residual signal. For example, the clipping results of the surrounding pixels in the residual signal are used as additional ALF fixed filter input.

[00423] In the fourth method, the spatial neighboring pixels right before SAO are used as additional ALF fixed filter inputs when training ALF fixed filters. Various filter shapes may be used to extract the information in the spatial neighboring pixels right before SAO. For example, the filter shape may be 1×1, 3×3 or 5×5 as shown in FIG. 15, or 13x13 diamond shape. Various equation forms may be used to extract the information in the spatial neighboring pixels right before SAO. For example, the clipping differences between the surrounding pixels right before SAO and current pixel are used as additional ALF fixed filter input.

[00424] In the fifth method, the methods presented in the first, second, third and fourth method may be combined. For example, both the spatial neighboring pixels right before deblocking filter and spatial neighboring pixels in residual signal are used as additional ALF fixed filter inputs when training ALF fixed filters.

[00425] ***Chroma ALF with sub-block level filter adaption***

[00426] According to the one or more embodiments of the disclosure, sub-block level filter adaption is applied in chroma ALF, where edge based classifier, band based classifier or residual based classifier are utilized in chroma ALF. Different methods may be used to achieve this goal.

[00427] In the first method, edge based classifier is utilized in chroma ALF. The edge based classifier may be defined in different sub-block levels, such as block size of 4x4, 2x2 or 1x1. The edge based classifier may be calculated in different methods. In the first example, the edge based classifier is calculated based on the reconstructed signal in luma component. For example, the edge based classifier which is utilized in luma ALF and calculated based on the right after SAO signal in luma component is reused for chroma ALF. In the second example, the edge based classifier is calculated based on the reconstructed signals in chroma components. For example, first one edge based classifier is calculated based on the right after SAO signal in Cb component, then another edge based classifier is calculated based on the right after SAO signal in Cr component. Finally, the edge based classifier which combines the first edge based classifier and the second edge based classifier is utilized for chroma ALF. It should be noted that when calculating the edge based classifier based on the right after SAO signal in Cb or Cr components, the calculating procedure may be same to calculating the edge based classifier based on the right after SAO signal in luma component. In the third example, the edge based classifier which combines the edge based classifier calculated in the first example and the edge based classifier calculated in the second example is utilized for chroma ALF.

[00428] It should be noted that when reusing the edge based classifier in luma ALF for chroma ALF, or recalculating the edge based classifier based on chroma components for chroma ALF, the specific class number utilized in chroma ALF may be same to or different from the class number utilized in luma ALF. For example, the class number utilized in luma ALF is 25, which is obtained by combining activity (5) and directionality (5), then the class number utilized in chroma ALF is 10, which is obtained by combining activity (2) and directionality (5).

[00429] In the second method, band based classifier is utilized in chroma ALF. The band based classifier may be defined in different sub-block levels, such as block size of 4x4, 2x2 or 1x1. The band based classifier may be calculated in different methods. In the first example, the band based classifier is calculated based on the reconstructed signal in luma component. For example,

the band based classifier which is utilized in luma ALF and calculated based on the right after SAO signal in luma component is reused for chroma ALF. In the second example, the band based classifier is calculated based on the reconstructed signals in chroma components. For example, first one band based classifier is calculated based on the right after SAO signal in Cb component, then another band based classifier is calculated based on the right after SAO signal in Cr component. Finally, the band based classifier which combines the first band based classifier and the second band based classifier is utilized for chroma ALF. It should be noted that when calculating the band based classifier based on the right after SAO signal in Cb or Cr components, the calculating procedure may be same to calculating the band based classifier based on the right after SAO signal in luma component. In the third example, the band based classifier which combines the band based classifier calculated in the first example and the band based classifier calculated in the second example is utilized for chroma ALF. It should be noted that when reusing the band based classifier in luma ALF for chroma ALF, or recalculating the band based classifier based on chroma components for chroma ALF, the specific class number utilized in chroma ALF may be same to or different from the class number utilized in luma ALF. For example, the class number utilized in luma ALF is 25, then the class number utilized in chroma ALF is 10.

[00430] In the third method, residual based classifier is utilized in chroma ALF. The residual based classifier may be defined in different sub-block levels, such as block size of 4x4, 2x2 or 1x1. The residual based classifier may be calculated in different methods. In the first example, the residual based classifier is calculated based on the residual signal in luma component. For example, the residual based classifier which is utilized in luma ALF and calculated based on the residual signal in luma component is reused for chroma ALF. In the second example, the residual based classifier is calculated based on the residual signals in chroma components. For example, first one residual based classifier is calculated based on the residual signal in Cb component, then another residual based classifier is calculated based on the residual signal in Cr component. Finally, the residual based classifier which combines the first residual based classifier and the second residual

based classifier is utilized for chroma ALF. It should be noted that when calculating the residual based classifier based on the residual signal in Cb or Cr components, the calculating procedure may be same to calculating the residual based classifier based on the residual signal in luma component. In the third example, the residual based classifier which combines the residual based classifier calculated in the first example and the residual based classifier calculated in the second example is utilized for chroma ALF. It should be noted that when reusing the residual based classifier in luma ALF for chroma ALF, or recalculating the residual based classifier based on chroma components for chroma ALF, the specific class number utilized in chroma ALF may be same to or different from the class number utilized in luma ALF. For example, the class number utilized in luma ALF is 25, then the class number utilized in chroma ALF is 10.

[00431] In the fourth method, the methods presented in the first, second and third method may be combined. For example, both the edge based classifier presented in the first method and the band based classifier presented in the second method are utilized in chroma ALF.

[00432] ***CCALF with sub-block level filter adaption***

[00433] According to the one or more embodiments of the disclosure, sub-block level filter adaption is applied in CCALF, where edge based classifier, band based classifier or residual based classifier are utilized in CCALF. Different methods may be used to achieve this goal.

[00434] In the first method, edge based classifier is utilized in CCALF. The edge based classifier may be defined in different sub-block levels, such as block size of 4x4, 2x2 or 1x1. The edge based classifier may be calculated in different methods. In the first example, the edge based classifier is calculated based on the reconstructed signal in luma component. For example, the edge based classifier which is utilized in luma ALF and calculated based on the right after SAO signal in luma component is reused for CCALF in both Cb and Cr components. In the second example, the edge based classifier is calculated based on the reconstructed signals in chroma components. For example, for CCALF in Cb component, the edge based classifier is calculated based on the right after SAO signal in Cb component; for CCALF in Cr component, the edge based classifier is

calculated based on the right after SAO signal in Cr component. It should be noted that when calculating the edge based classifier based on the right after SAO signal in Cb or Cr components, the calculating procedure may be same to calculating the edge based classifier based on the right after SAO signal in luma component. In the third example, the edge based classifier which combines the edge based classifier calculated in the first example and the edge based classifier calculated for CCALF in Cb component in the second example is utilized for CCALF in Cb component; the edge based classifier which combines the edge based classifier calculated in the first example and the edge based classifier calculated for CCALF in Cr component in the second example is utilized for CCALF in Cr component.

[00435] It should be noted that when reusing the edge based classifier in luma ALF for CCALF, or recalculating the edge based classifier based on chroma components for CCALF, the specific class number utilized in CCALF may be same to or different from the class number utilized in luma ALF. For example, the class number utilized in luma ALF is 25, which is obtained by combining activity (5) and directionality (5), then the class number utilized in CCALF is 10, which is obtained by combining activity (2) and directionality (5).

[00436] In the second method, band based classifier is utilized in CCALF. The band based classifier may be defined in different sub-block levels, such as block size of 4x4, 2x2 or 1x1. The band based classifier may be calculated in different methods. In the first example, the band based classifier is calculated based on the reconstructed signal in luma component. For example, the band based classifier which is utilized in luma ALF and calculated based on the right after SAO signal in luma component is reused for CCALF in both Cb and Cr components. In the second example, the band based classifier is calculated based on the reconstructed signals in chroma components. For example, for CCALF in Cb component, the band based classifier is calculated based on the right after SAO signal in Cb component; for CCALF in Cr component, the band based classifier is calculated based on the right after SAO signal in Cr component. It should be noted that when calculating the band based classifier based on the right after SAO signal in Cb or Cr components,

the calculating procedure may be same to calculating the band based classifier based on the right after SAO signal in luma component. In the third example, the band based classifier which combines the band based classifier calculated in the first example and the band based classifier calculated for CCALF in Cb component in the second example is utilized for CCALF in Cb component; the band based classifier which combines the band based classifier calculated in the first example and the band based classifier calculated for CCALF in Cr component in the second example is utilized for CCALF in Cr component. It should be noted that when reusing the band based classifier in luma ALF for CCALF, or recalculating the band based classifier based on chroma components for CCALF, the specific class number utilized in CCALF may be same to or different from the class number utilized in luma ALF. For example, the class number utilized in luma ALF is 25, then the class number utilized in CCALF is 10.

[00437] In the third method, residual based classifier is utilized in CCALF. The residual based classifier may be defined in different sub-block levels, such as block size of 4x4, 2x2 or 1x1. The residual based classifier may be calculated in different methods. In the first example, the residual based classifier is calculated based on the residual signal in luma component. For example, the residual based classifier which is utilized in luma ALF and calculated based on the residual signal in luma component is reused for CCALF in both Cb and Cr components. In the second example, the residual based classifier is calculated based on the residual signals in chroma components. For example, for CCALF in Cb component, the residual based classifier is calculated based on the residual signal in Cb component; for CCALF in Cr component, the residual based classifier is calculated based on the residual signal in Cr component. It should be noted that when calculating the residual based classifier based on the residual signal in Cb or Cr components, the calculating procedure may be same to calculating the residual based classifier based on the residual signal in luma component. In the third example, the residual based classifier which combines the residual based classifier calculated in the first example and the residual based classifier calculated for CCALF in Cb component in the second example is utilized for CCALF in Cb component; the

residual based classifier which combines the residual based classifier calculated in the first example and the residual based classifier calculated for CCALF in Cr component in the second example is utilized for CCALF in Cr component. It should be noted that when reusing the residual based classifier in luma ALF for CCALF, or recalculating the residual based classifier based on chroma components for CCALF, the specific class number utilized in CCALF may be same to or different from the class number utilized in luma ALF. For example, the class number utilized in luma ALF is 25, then the class number utilized in CCALF is 10.

[00438] In the fourth method, the methods presented in the first, second and third method may be combined. For example, both the edge based classifier presented in the first method and the band based classifier presented in the second method are utilized in CCALF.

[00439] FIG. 18 shows a computing environment 1810 coupled with a user interface 1850. The computing environment 1810 can be part of a data processing server. The computing environment 1810 includes a processor 1820, a memory 1830, and an Input/Output (I/O) interface 1840.

[00440] The processor 1820 typically controls overall operations of the computing environment 1810, 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.

[00441] The memory 1830 is configured to store various types of data to support the operation of the computing environment 1810. The memory 1830 may include predetermined software 1832. Examples of such data includes instructions for any applications or methods operated on the computing environment 1810, 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.

[00442] 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 and decoder.

[00443] 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 1810, 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 an embodiment, the plurality of programs may be executed by the processor 1820 in the computing environment 1810 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 1810 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 1810 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 1810 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.

[00444] 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.

[00445] 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.

[00446] 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 1810, for performing the above-described methods. For example, the computer program product may include the non-transitory computer-readable storage medium.

[00447] In an embodiment, the computing environment 1810 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.

[00448] FIG. 19 is a flowchart illustrating a method for video decoding in accordance with some examples of the present disclosure. At step 1901, the processor 1820, at the side of a decoder, may obtain an adaptive loop filter (ALF) classifier for a sub-block. In some embodiments, the ALF classifier may be an edge-based, band-based, or residual-based classifier selected based on predefined criteria or the characteristics of the video content. Each classifier type offers a different approach to determining how the ALF will be applied to enhance chroma quality. In some embodiments, an ALF classifier used for filtering a luma sample may be used as the ALF for enhancing chroma quality, or the ALF classifier may be calculated based on the features of the chroma samples. At step 1902, the processor 1820 may determine an ALF based on the ALF classifier. The ALF classifier analyzes the picture to categorize different sub-blocks based on their features. For each classified sub-block, a filter is specifically determined for that class. At step 1903, the processor 1820 may obtain a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample. The filtering process based on the ALF outputs a filtered chroma sample with an enhanced chroma representation, reducing noise or artifacts, enhancing edge clarity, or addressing residual discrepancies.

[00449] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a luma ALF classifier as the ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

[00450] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample; obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample; and deriving a combined classifier as the ALF classifier based on the first feature and the second feature.

[00451] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block; obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample; obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample; deriving a combined classifier based on the first feature and the second feature; and deriving the ALF classifier based on the luma ALF and the combined classifier.

[00452] In some embodiments, a class number of the ALF classifier is less than a class number of a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block.

[00453] In some embodiments, the class number of the ALF classifier is 10.

[00454] In some embodiments, the ALF includes a cross component adaptive loop filter (CCALF).

[00455] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a luma ALF classifier as the ALF classifier for both first and second chroma components, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

[00456] In some embodiments, obtaining the ALF classifier for a sub-block includes one of following operations: obtaining a first ALF classifier based on a first signal in a first chroma component; or obtaining a second ALF classifier based on a second signal in a second chroma component.

[00457] In some embodiments, obtaining the ALF classifier for a sub-block includes one of following operations: obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a first ALF classifier based on a first signal in a first chroma component; and obtaining a combined ALF classifier as the ALF classifier for the first chroma component based on the luma ALF classifier and the first ALF classifier; or, obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma

sample in the sub-block; obtaining a second ALF classifier based on a second signal in a second chroma component; and obtaining a combined ALF classifier as the ALF classifier for the second chroma component based on the luma ALF classifier and the second ALF classifier.

[00458] In some embodiments, the ALF classifier includes any one or any combination of an ALF classifier that is edge-based, an ALF classifier that is band based, or an ALF classifier that is residual based.

[00459] In some embodiments, a block size of the sub-block includes any one of 4x4, 2x2 or 1x1.

[00460] FIG. 20 is a flowchart illustrating a method for video encoding in accordance with some examples of the present disclosure. At step 2001, the processor 1820, at the side of an encoder, may obtain an adaptive loop filter (ALF) classifier for a sub-block. In some embodiments, the ALF classifier may be an edge-based, band-based, or residual-based classifier selected based on predefined criteria or the characteristics of the video content. Each classifier type offers a different approach to determining how the ALF will be applied to enhance chroma quality. In some embodiments, an ALF classifier used for filtering a luma sample may be used as the ALF for enhancing chroma quality, or the ALF classifier may be calculated based on the features of the chroma samples. At step 2002, the processor 1820 may determine an ALF based on the ALF classifier. The ALF classifier analyzes the picture to categorize different sub-blocks based on their features. For each classified sub-block, a filter is specifically determined for that class. At step 2003, the processor 1820 may obtain a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample. The filtering process based on the ALF outputs a filtered chroma sample with an enhanced chroma representation, reducing noise or artifacts, enhancing edge clarity, or addressing residual discrepancies.

[00461] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a luma ALF classifier as the ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

[00462] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample; obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample; and deriving a combined classifier as the ALF classifier based on the first feature and the second feature.

[00463] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block; obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample; obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample; deriving a combined classifier based on the first feature and the second feature; and deriving the ALF classifier based on the luma ALF and the combined classifier.

[00464] In some embodiments, a class number of the ALF classifier is less than a class number of a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block.

[00465] In some embodiments, the class number of the ALF classifier is 10.

[00466] In some embodiments, the ALF includes a cross component adaptive loop filter (CCALF).

[00467] In some embodiments, obtaining the ALF classifier for the sub-block includes: obtaining a luma ALF classifier as the ALF classifier for both first and second chroma components, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

[00468] In some embodiments, obtaining the ALF classifier for a sub-block includes one of following operations: obtaining a first ALF classifier based on a first signal in a first chroma component; or obtaining a second ALF classifier based on a second signal in a second chroma component.

[00469] In some embodiments, obtaining the ALF classifier for a sub-block includes one

of following operations: obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a first ALF classifier based on a first signal in a first chroma component; and obtaining a combined ALF classifier as the ALF classifier for the first chroma component based on the luma ALF classifier and the first ALF classifier; or, obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a second ALF classifier based on a second signal in a second chroma component; and obtaining a combined ALF classifier as the ALF classifier for the second chroma component based on the luma ALF classifier and the second ALF classifier.

[00470] In some embodiments, the ALF classifier includes any one or any combination of an ALF classifier that is edge-based, an ALF classifier that is band based, or an ALF classifier that is residual based.

[00471] In some embodiments, a block size of the sub-block includes any one of 4x4, 2x2 or 1x1.

[00472] 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 encoded by the encoding method described above.

[00473] 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 encoded by the encoder described above.

[00474] 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.

[00475] 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.

[00476] 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.

WHAT IS CLAIMED IS:

1. A method for video decoding, comprising:

obtaining, by a decoder, an adaptive loop filter (ALF) classifier for a sub-block;

determining, by the decoder, an ALF based on the ALF classifier; and

obtaining, by the decoder, a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample.

2. The method of claim 1, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a luma ALF classifier as the ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

3. The method of claim 1, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample;

obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample; and

deriving a combined classifier as the ALF classifier based on the first feature and the second feature.

4. The method of claim 1, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block;

obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample;

obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample;

deriving a combined classifier based on the first feature and the second feature; and

deriving the ALF classifier based on the luma ALF and the combined classifier.

5. The method of claim 1, wherein a class number of the ALF classifier is less than a class number of a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block.

6. The method of claim 5, wherein the class number of the ALF classifier is 10.

7. The method of claim 1, wherein the ALF comprises a cross component adaptive loop filter (CCALF).

8. The method of claim 7, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a luma ALF classifier as the ALF classifier for both first and second chroma components, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

9. The method of claim 7, wherein obtaining the ALF classifier for a sub-block comprises one of following operations:

obtaining a first ALF classifier based on a first signal in a first chroma component; or

obtaining a second ALF classifier based on a second signal in a second chroma component.

10. The method of claim 7, wherein obtaining the ALF classifier for a sub-block comprises one of following operations:

obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a first ALF classifier based on a first signal in a first chroma component; and obtaining a combined ALF classifier as the ALF classifier for the first chroma component based on the luma ALF classifier and the first ALF classifier;

or,

obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a second ALF classifier based on a second signal in a second chroma component; and obtaining a combined ALF classifier as the ALF classifier for the second chroma component based on the luma ALF classifier and the second ALF classifier.

11. The method of claim 1, wherein the ALF classifier comprises any one or any combination of an ALF classifier that is edge-based, an ALF classifier that is band based, or an ALF classifier that is residual based.

12. The method of claim 1, wherein a block size of the sub-block comprises any one of 4x4, 2x2 or 1x1.

13. A method for video encoding, comprising:

obtaining, by an encoder, an adaptive loop filter (ALF) classifier for a sub-block;

determining, by the encoder, an ALF based on the ALF classifier; and

obtaining, by the encoder, a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample.

14. The method of claim 13, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a luma ALF classifier as the ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

15. The method of claim 13, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample;

obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample; and

deriving a combined classifier as the ALF classifier based on the first feature and the second feature.

16. The method of claim 13, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block;

obtaining a first feature based on a first ALF classifier and a first signal in a first chroma component of the chroma sample;

obtaining a second feature based on a second ALF classifier and a second signal in a second chroma component of the chroma sample;

deriving a combined classifier based on the first feature and the second feature; and

deriving the ALF classifier based on the luma ALF and the combined classifier.

17. The method of claim 13, wherein a class number of the ALF classifier is less than a class number of a luma ALF classifier used for determining a luma ALF for a luma sample in the sub-block.

18. The method of claim 17, wherein the class number of the ALF classifier is 10.

19. The method of claim 13, wherein the ALF comprises a cross component adaptive loop filter (CCALF).

20. The method of claim 19, wherein obtaining the ALF classifier for the sub-block comprises:

obtaining a luma ALF classifier as the ALF classifier for both first and second chroma components, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block.

21. The method of claim 19, wherein obtaining the ALF classifier for a sub-block comprises one of following operations:

obtaining a first ALF classifier based on a first signal in a first chroma component; or

obtaining a second ALF classifier based on a second signal in a second chroma component.

22. The method of claim 19, wherein obtaining the ALF classifier for a sub-block comprises one of following operations:

obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a first ALF classifier based on a first signal in a first chroma component; and obtaining a combined ALF classifier as the ALF classifier for the first chroma component based on the luma ALF classifier and the first ALF classifier;

or,

obtaining a luma ALF classifier, the luma ALF being used for determining a luma ALF for a luma sample in the sub-block; obtaining a second ALF classifier based on a second signal in a second chroma component; and obtaining a combined ALF classifier as the ALF classifier for the second chroma component based on the luma ALF classifier and the second ALF classifier.

23. The method of claim 13, wherein the ALF classifier comprises any one or any combination of an ALF classifier that is edge-based, an ALF classifier that is band based, or an ALF classifier that is residual based.

24. The method of claim 13, wherein a block size of the sub-block comprises any one of 4x4, 2x2 or 1x1.

25. An apparatus for video decoding, comprising:

one or more processors; and

a memory coupled to the one or more processors and configured to store instructions executable by the one or more processors,

wherein the one or more processors, upon execution of the instructions, are configured to perform the method in any one of claims 1-12.

26. A non-transitory computer-readable storage medium for storing computer-executable instructions that, when executed by one or more computer processors, cause the one or more computer processors to perform the method in any of claims 1-12.

27. An apparatus for video encoding, comprising:

one or more processors; and

a memory coupled to the one or more processors and configured to store instructions executable by the one or more processors,

wherein the one or more processors, upon execution of the instructions, are configured to perform the method in any one of claims 13-24.

28. A non-transitory computer-readable storage medium for storing computer-executable instructions that, when executed by one or more computer processors, cause the one or more computer processors to perform the method in any of claims 13-24.

29. A non-transitory computer-readable storage medium for storing a bitstream to be decoded by the method in any of claims 1-12.

30. A non-transitory computer-readable storage medium for storing a bitstream generated by the method in any of claims 13-24.

31. A method for receiving a bitstream, wherein the bitstream comprises encoded video information to be decoded by the method in any one of claims 1-12.

32. A method for transmitting a bitstream, wherein the bitstream comprises encoded video information generated by the method of claims 13-24.

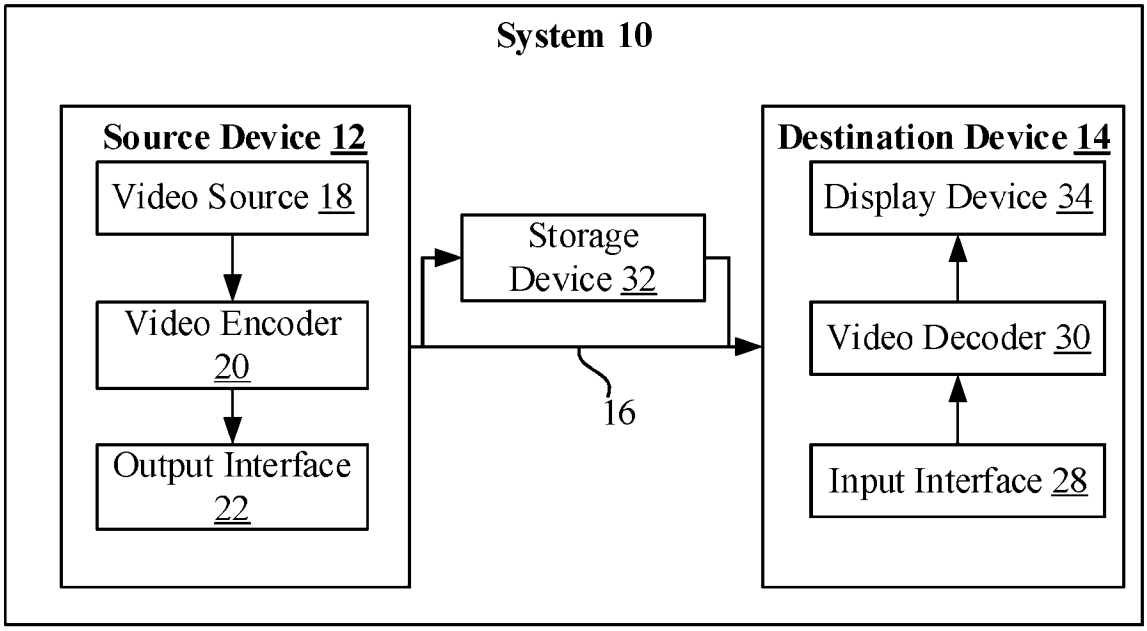

System 10
Source Device 12
Video Source 18
Video Encoder 20
Output Interface 22
Storage Device 32
16
Destination Device 14
Display Device 34
Video Decoder 30
Input Interface 28

FIG.1

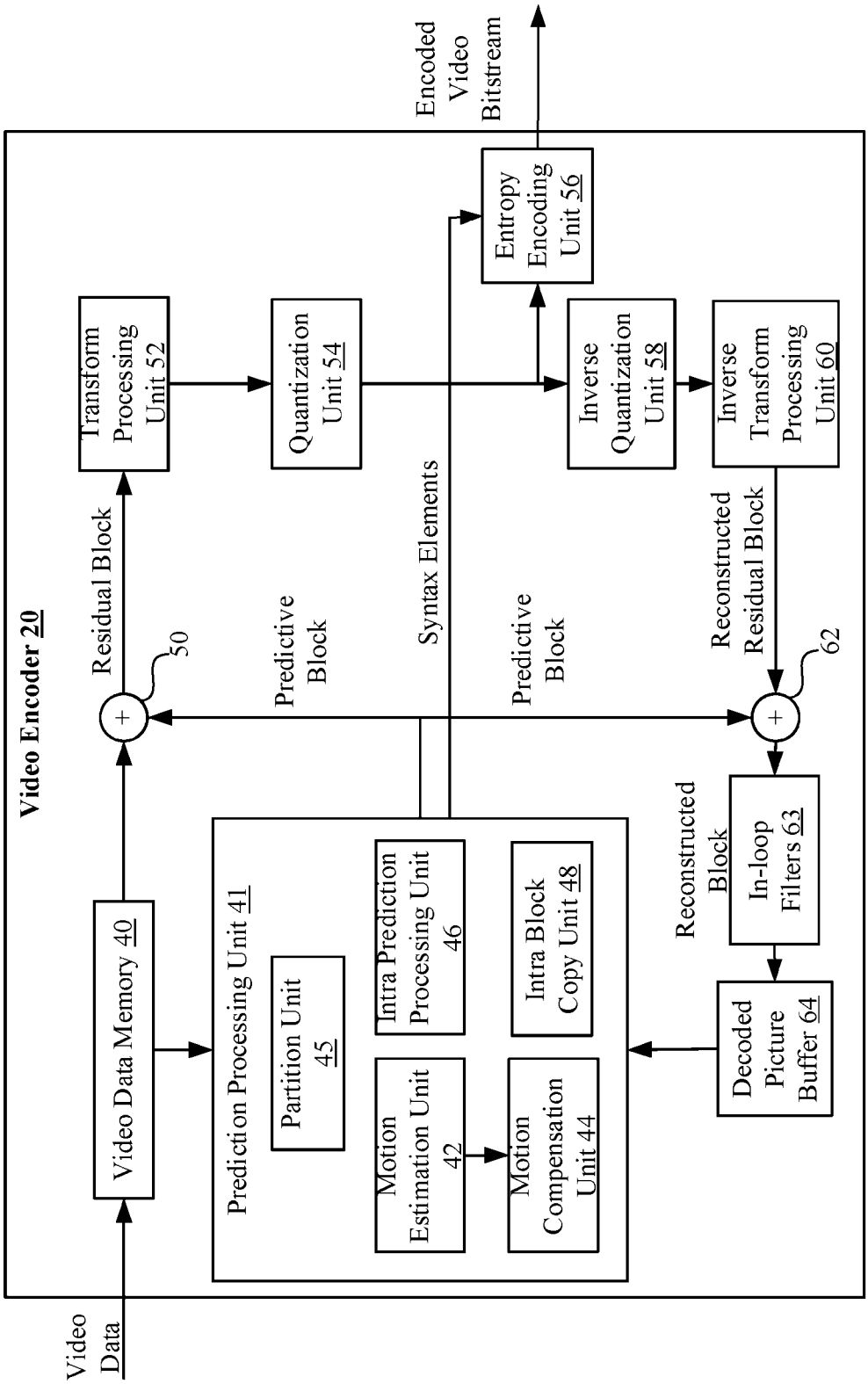

Video Encoder 20
Video Data
Video Data Memory 40
Prediction Processing Unit 41
Partition Unit 45
Motion Estimation Unit 42
Motion Compensation Unit 44
Intra Prediction Processing Unit 46
Intra Block Copy Unit 48
Residual Block
50
Transform Processing Unit 52
Quantization Unit 54
Predictive Block
Syntax Elements
Predictive Block
Entropy Encoding Unit 56
Encoded Video Bitstream
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

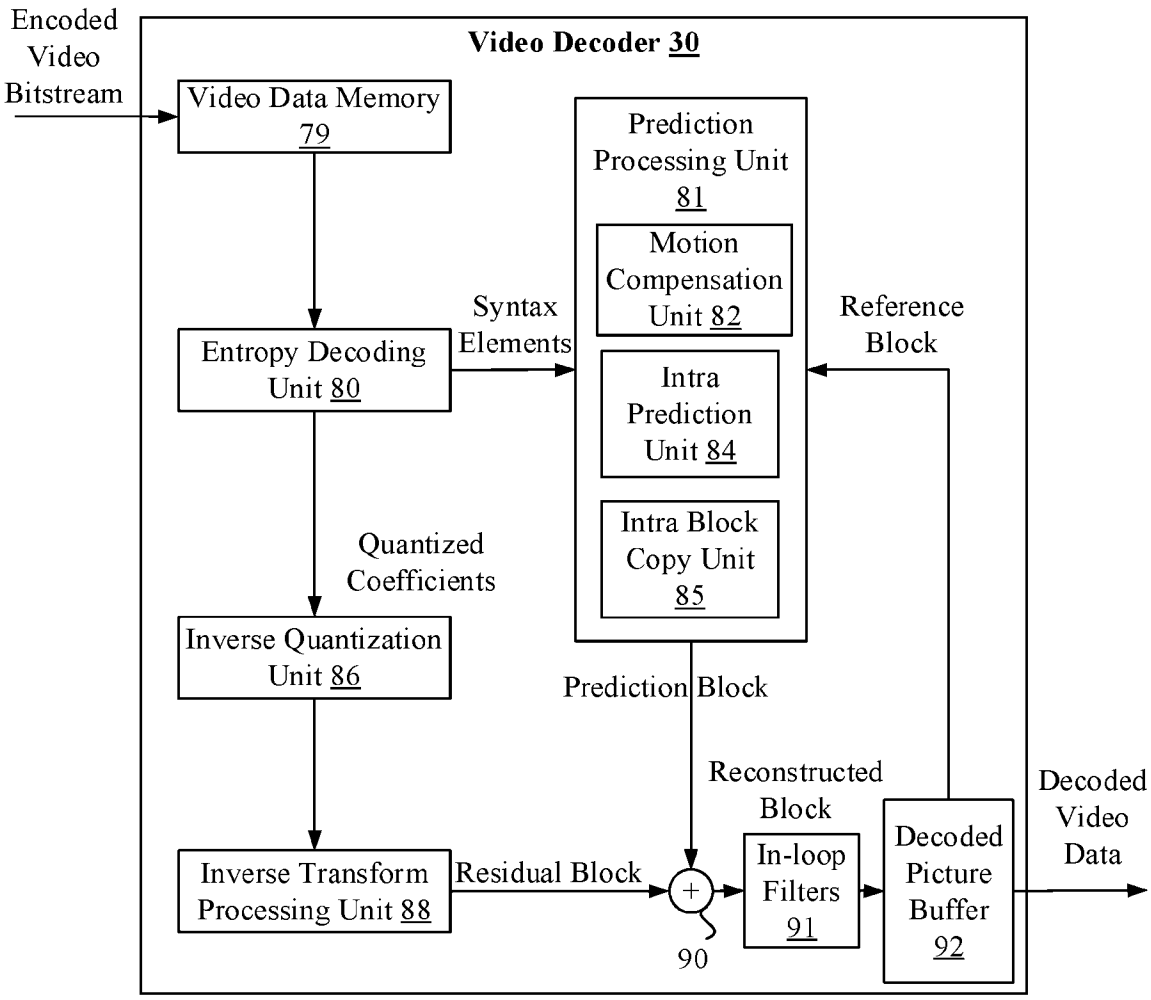

Encoded Video Bitstream
Video Decoder 30
Video Data Memory 79
Prediction Processing Unit 81
Motion Compensation Unit 82
Syntax Elements
Reference Block
Entropy Decoding Unit 80
Intra Prediction Unit 84
Intra Block Copy Unit 85
Quantized Coefficients
Inverse Quantization Unit 86
Prediction Block
Reconstructed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

FIG.3

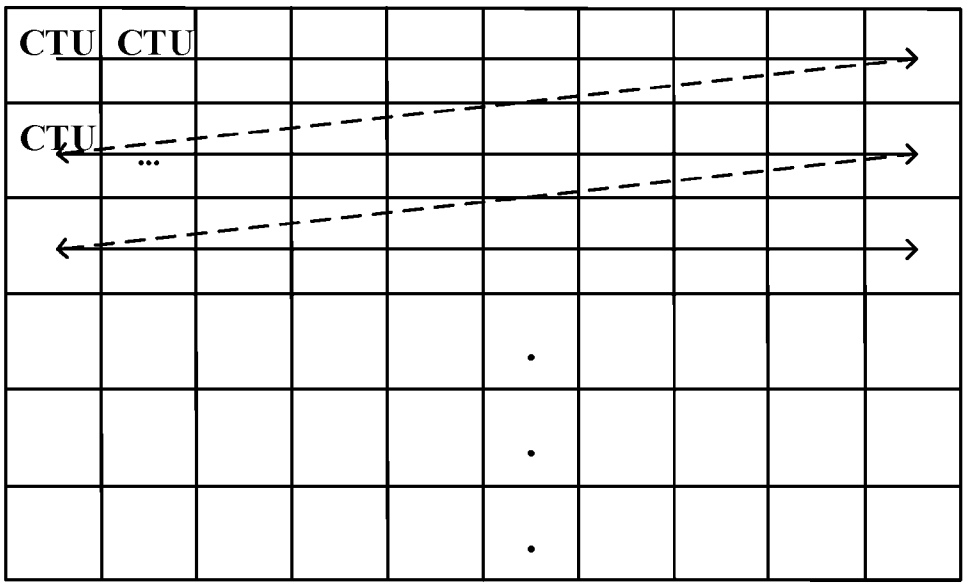

CTU
CTU
CTU
...

FIG.4A

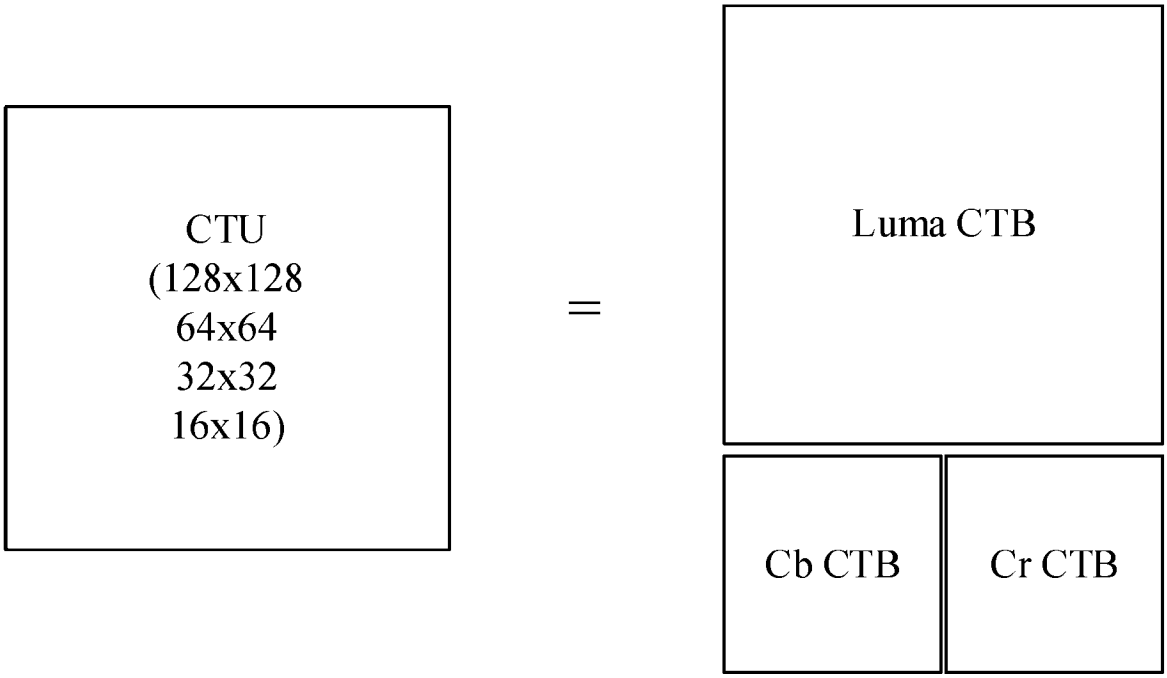

CTU
(128x128
64x64
32x32
16x16)
=
Luma CTB
Cb CTB
Cr CTB

FIG.4B

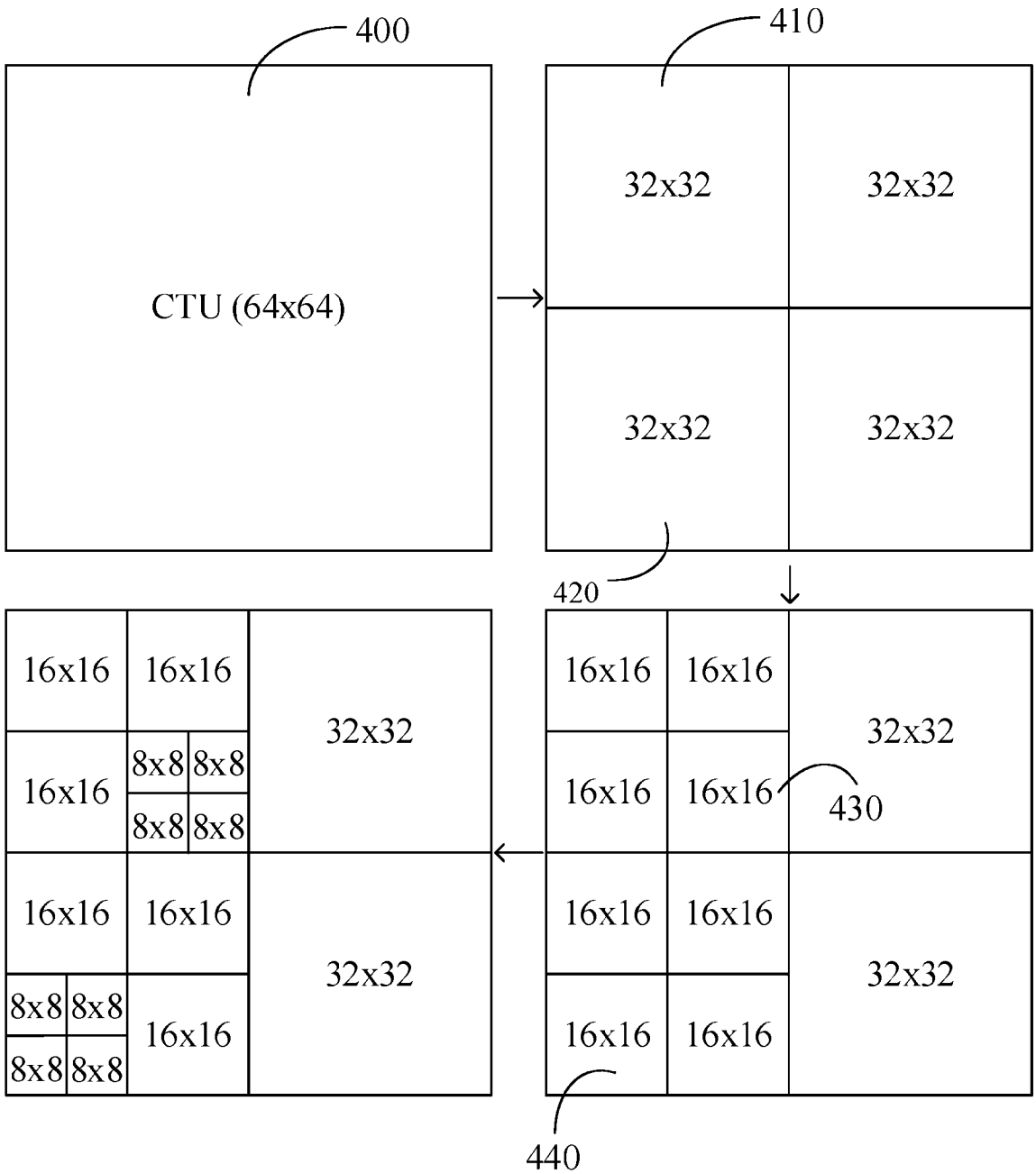

400
CTU (64x64)
410
32x32
32x32
32x32
32x32
420
16x16
16x16
32x32
16x16
16x16
430
16x16
16x16
32x32
16x16
16x16
32x32
16x16
16x16
8x8
8x8
8x8
8x8
16x16
8x8
8x8
8x8
8x8
16x16
440

FIG.4C

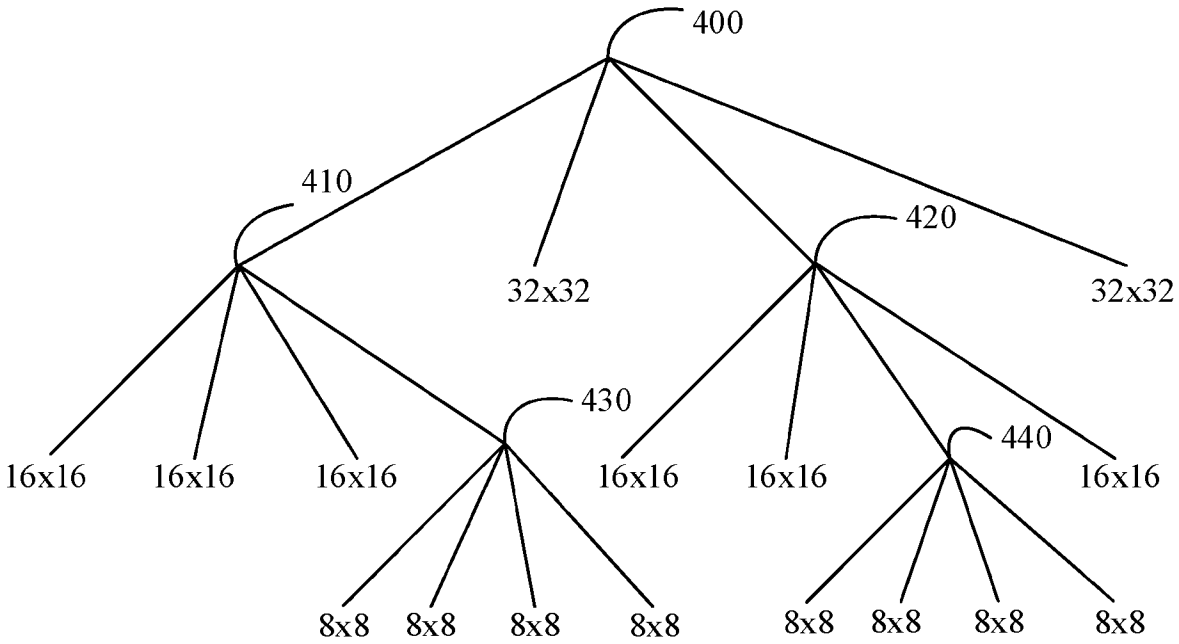

400
410
420
32x32
32x32
430
440
16x16
16x16
16x16
16x16
16x16
16x16
8x8
8x8
8x8
8x8
8x8
8x8
8x8
8x8

FIG.4D

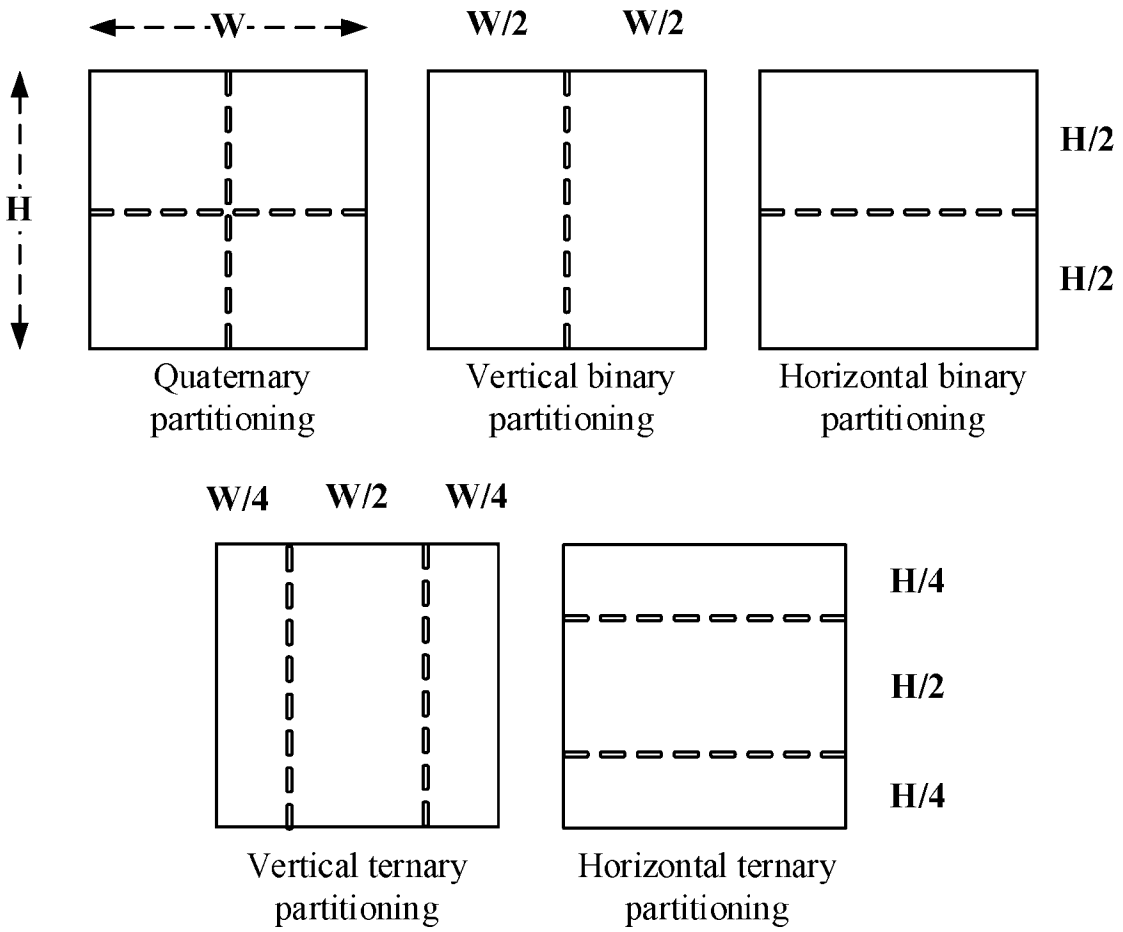

W
H
W/2
W/2
H/2
H/2
Quaternary partitioning
Vertical binary partitioning
Horizontal binary partitioning
W/4
W/2
W/4
H/4
H/2
H/4
Vertical ternary partitioning
Horizontal ternary partitioning

FIG.4E

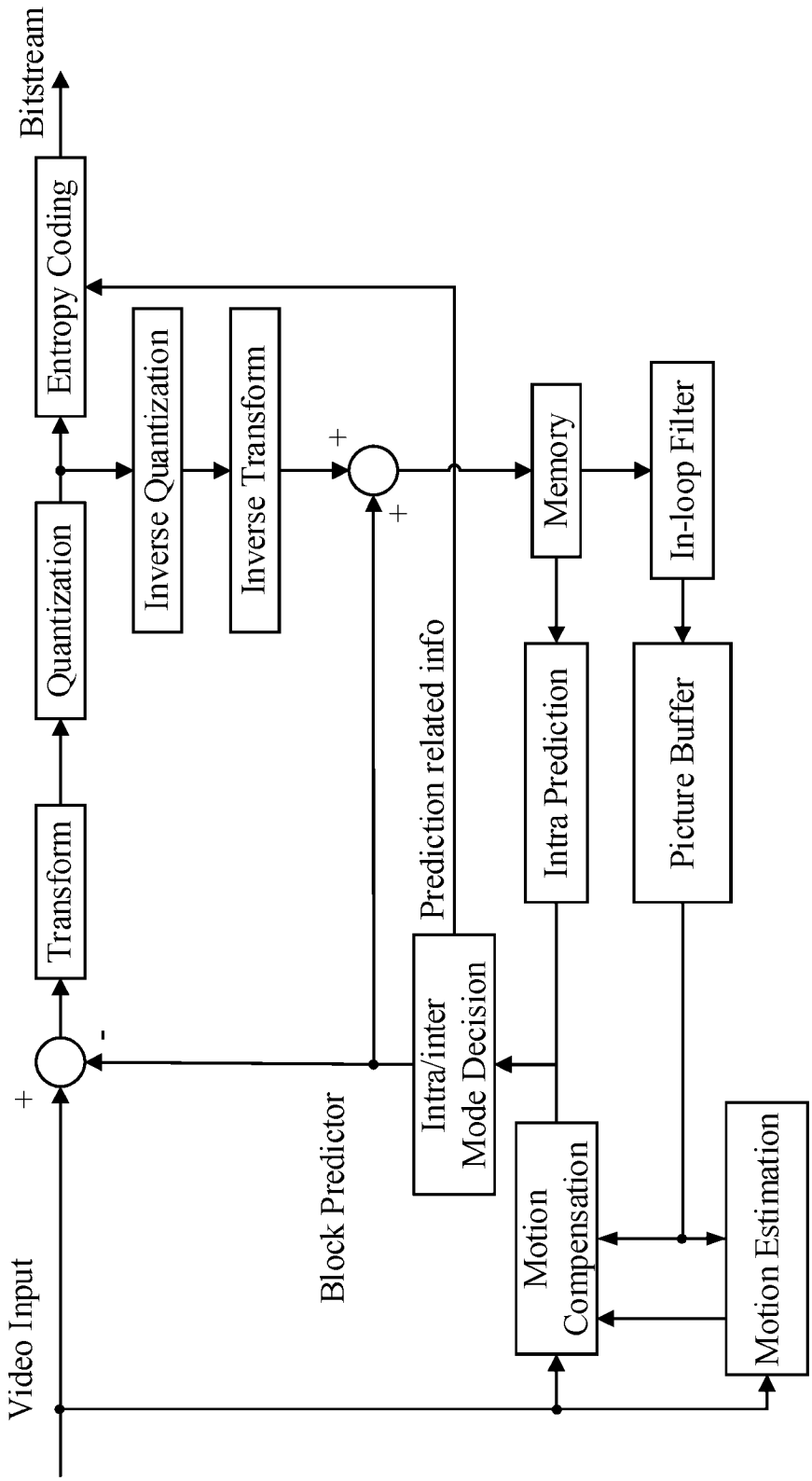

Video Input
+
-
Transform
Quantization
Entropy Coding
Bitstream
Block Predictor
Inverse Quantization
Inverse Transform
+
+
Intra/inter
Mode Decision
Prediction related info
Memory
Intra Prediction
Motion
Compensation
In-loop Filter
Picture Buffer
Motion Estimation

FIG. 5

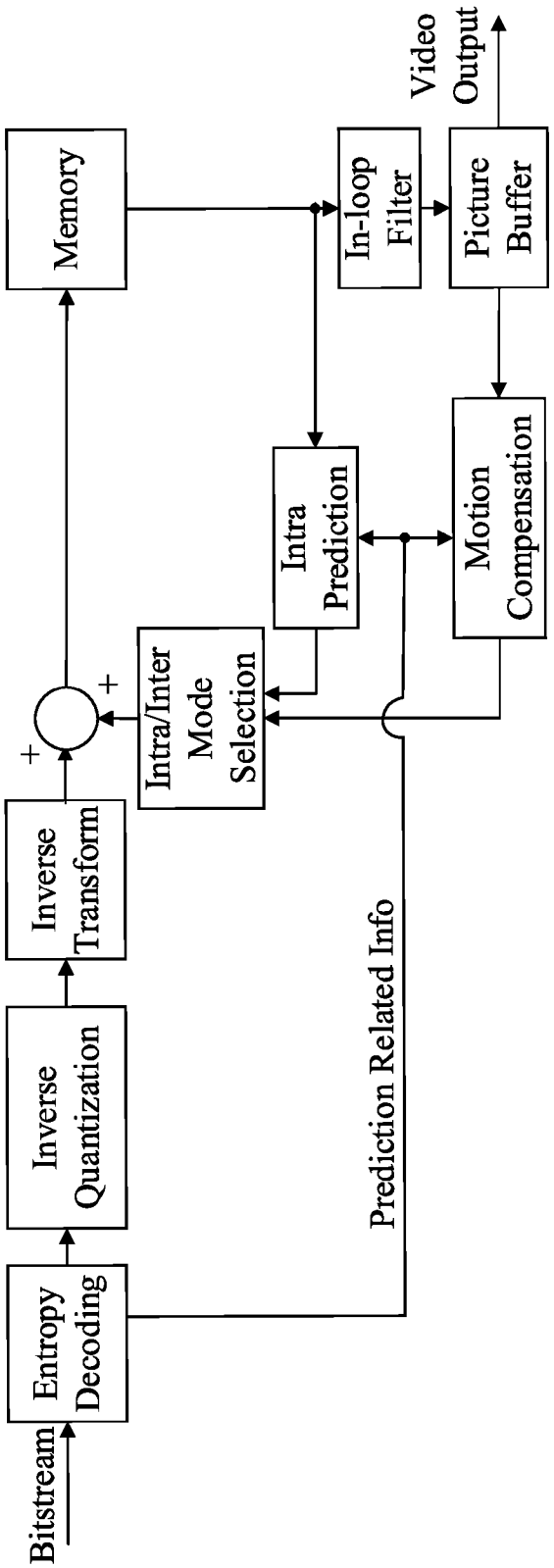

Bitstream
Entropy Decoding
Inverse Quantization
Inverse Transform
+
+
Memory
Intra/Inter Mode Selection
Intra Prediction
Prediction Related Info
Motion Compensation
In-loop Filter
Picture Buffer
Video Output

FIG. 6

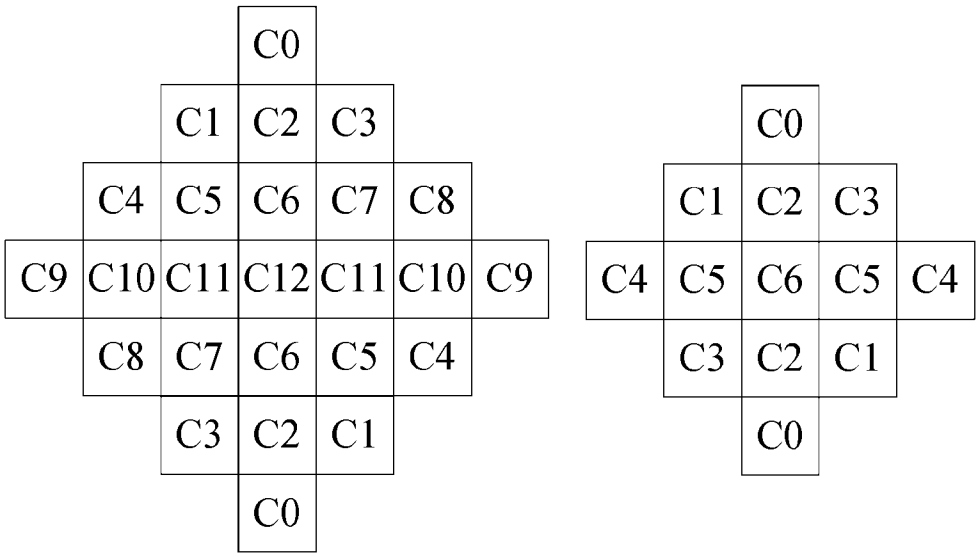

C0
C1 C2 C3
C4 C5 C6 C7 C8
C9 C10 C11 C12 C11 C10 C9
C8 C7 C6 C5 C4
C3 C2 C1
C0
C0
C1 C2 C3
C4 C5 C6 C5 C4
C3 C2 C1
C0

FIG.7

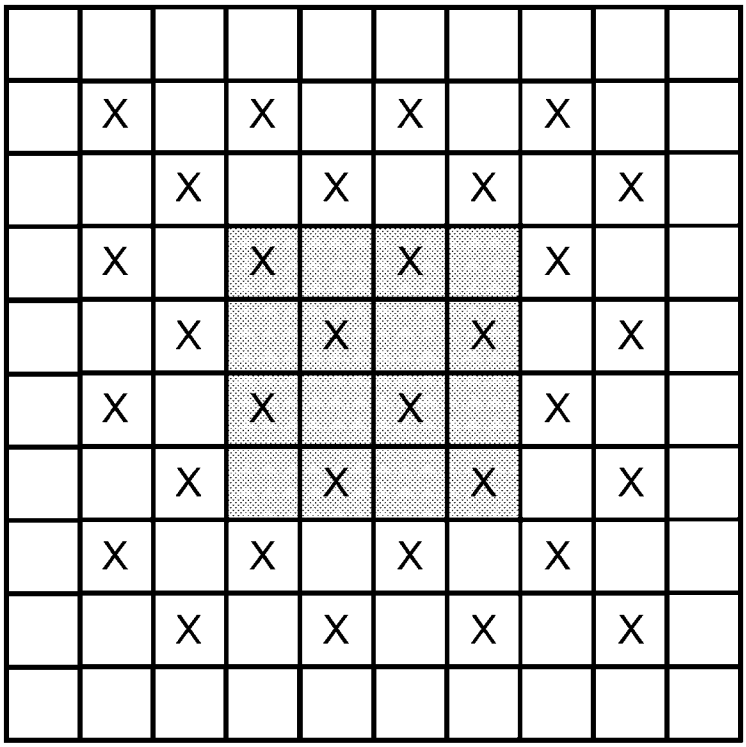


FIG.8

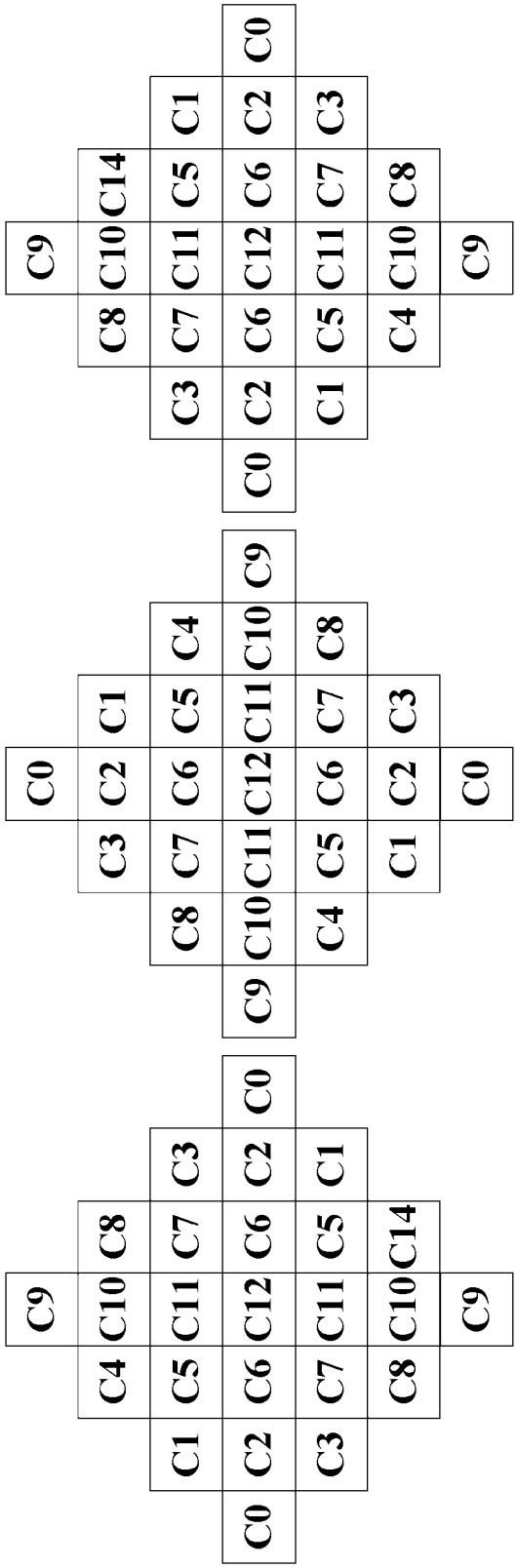

C0
C1 C2 C3
C14 C5 C6 C7 C8
C9 C10 C11 C12 C11 C10 C9
C8 C7 C6 C5 C4
C3 C2 C1
C0
C9
C4 C10 C8
C1 C5 C11 C7 C3
C0 C2 C6 C12 C6 C2 C0
C3 C7 C11 C5 C1
C8 C10 C4
C9
C0
C3 C2 C1
C8 C7 C6 C5 C14
C9 C10 C11 C12 C11 C10 C9
C4 C5 C6 C7 C8
C1 C2 C3
C0

FIG. 9

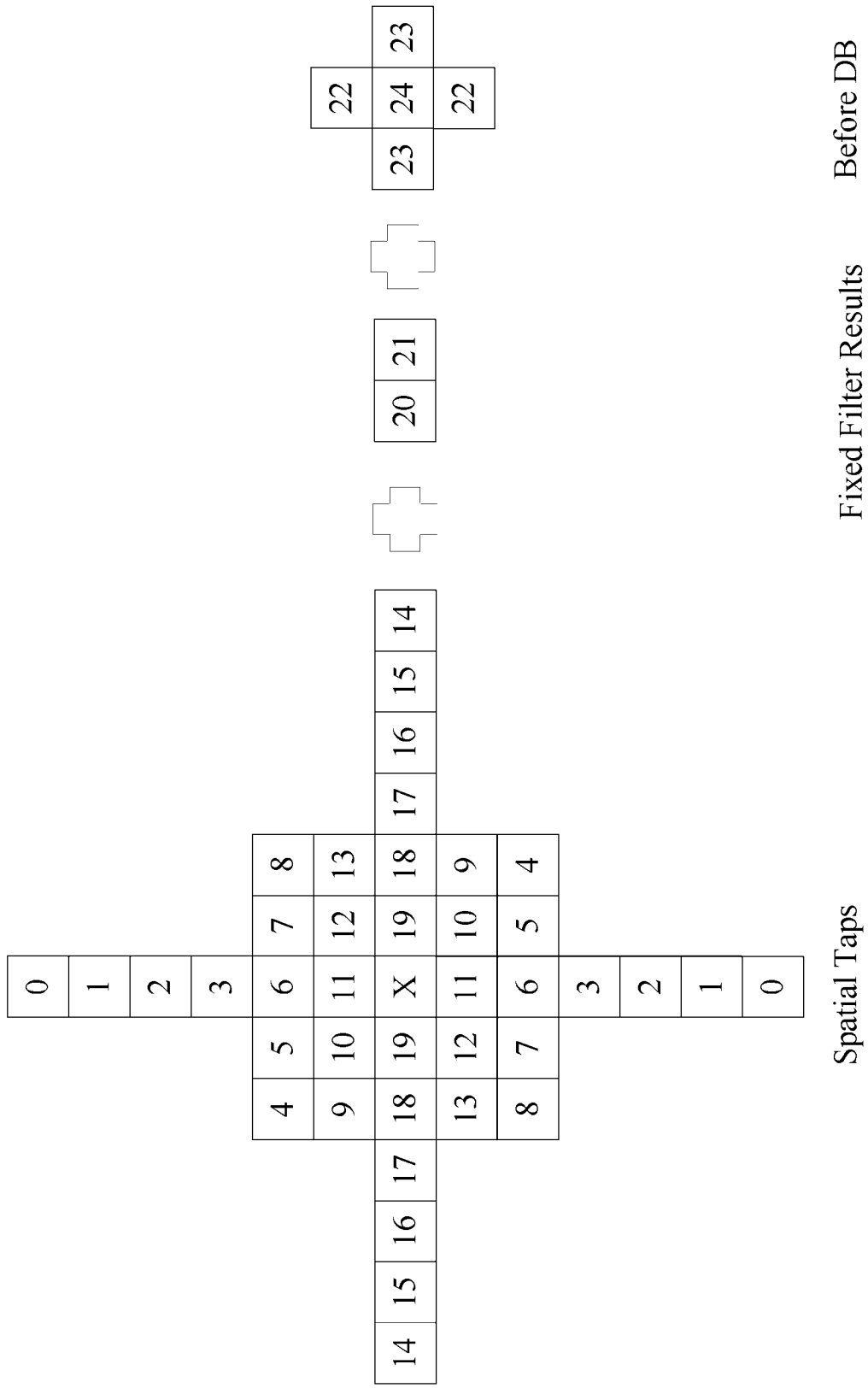

Spatial Taps
Fixed Filter Results
Before DB
0 1 2 3 6 11 X 11 6 3 2 1 0
5 10 19 12 7
4 9 18 13 8
7 12 19 10 5
8 13 18 9 4
14 15 16 17
17 16 15 14
20 21
23 24 22
22 23

FIG. 10

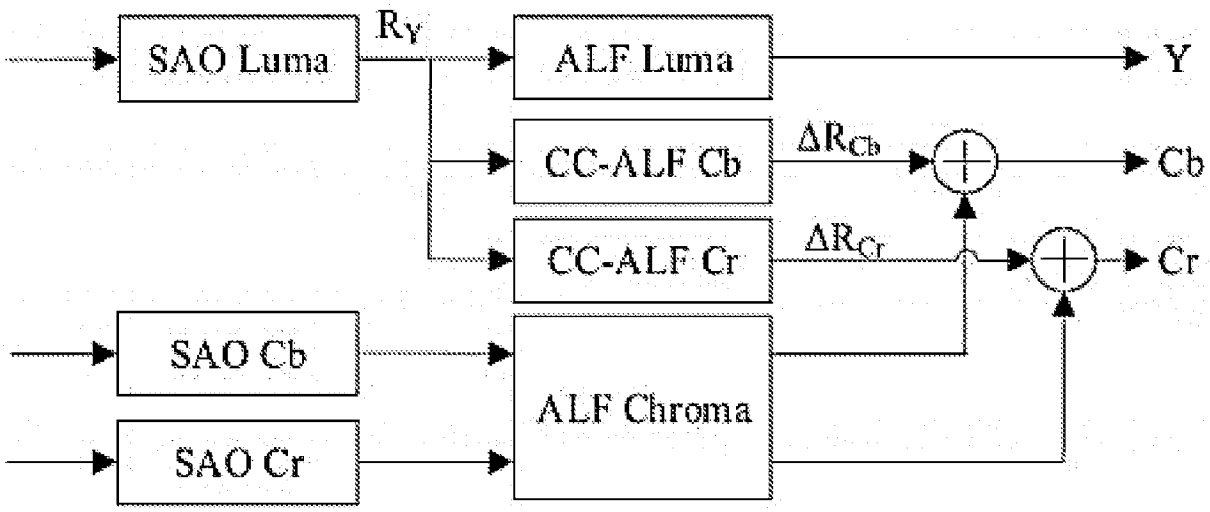

SAO Luma
R_Y
ALF Luma
Y
CC-ALF Cb
ΔR_{Cb}
Cb
CC-ALF Cr
ΔR_{Cr}
Cr
SAO Cb
SAO Cr
ALF Chroma

FIG.11

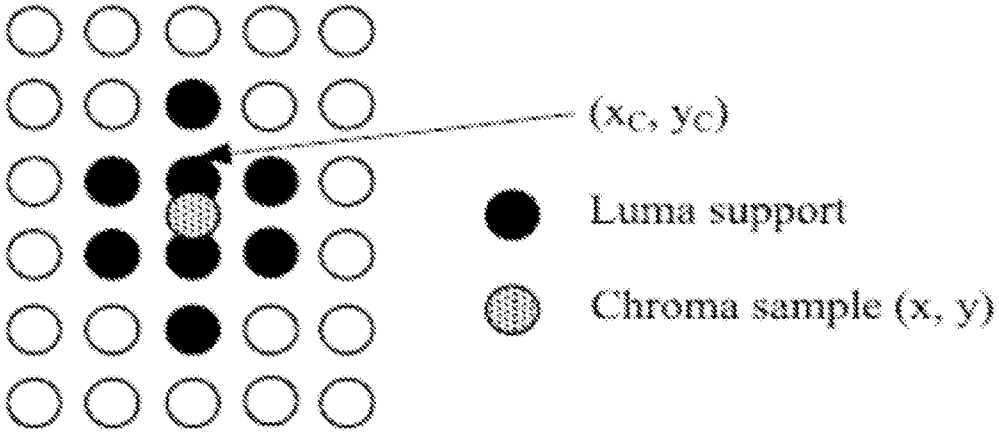

(x_C, y_C)
Luma support
Chroma sample (x, y)

FIG. 12

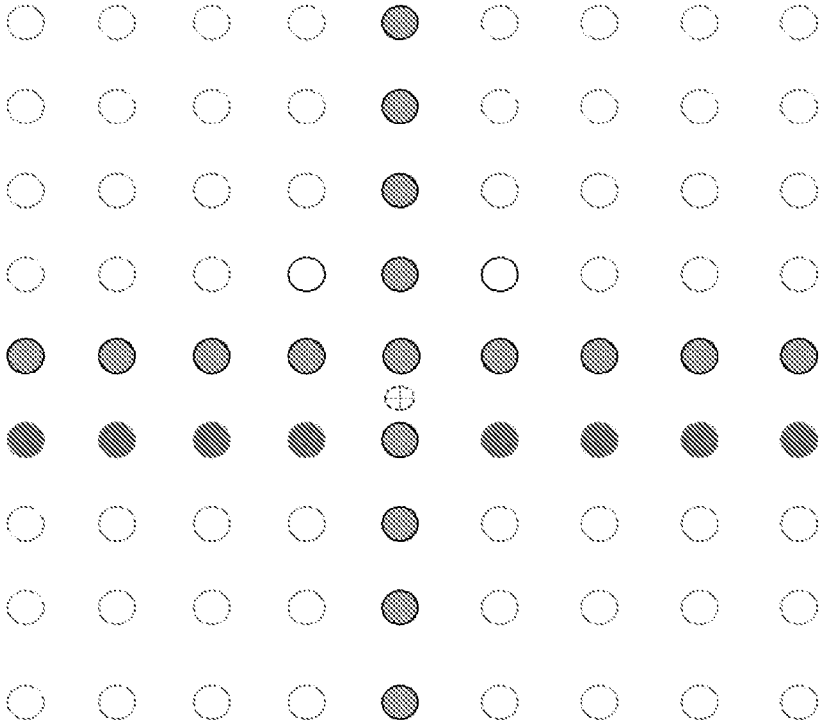


FIG. 13

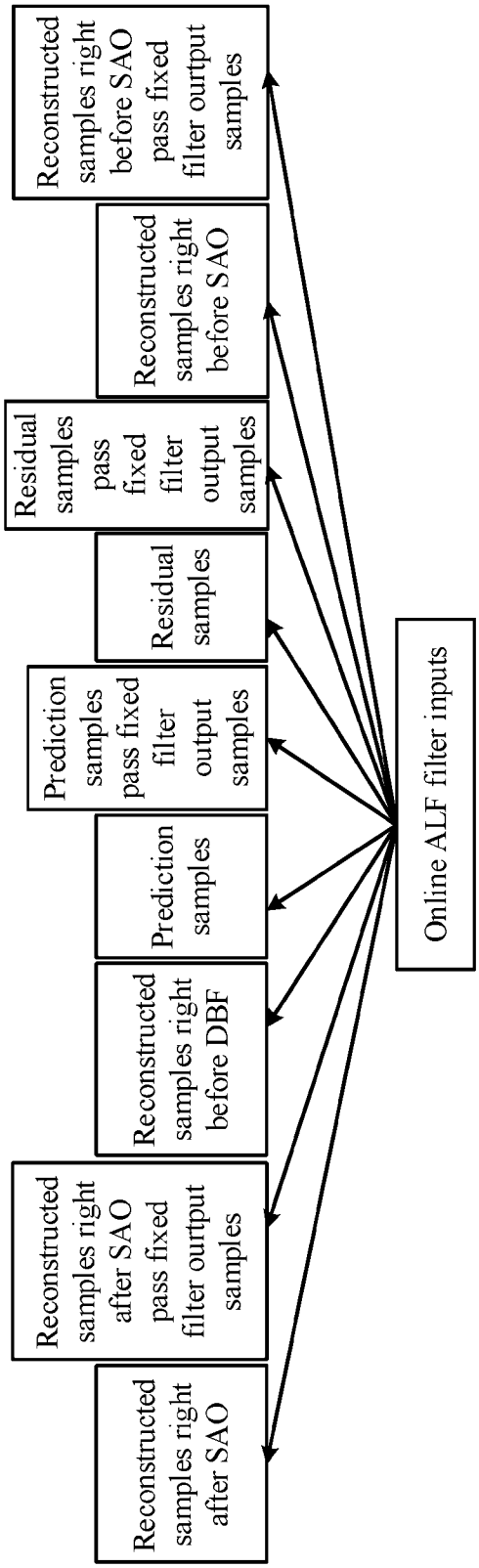

Reconstructed samples right after SAO
Reconstructed samples right after SAO pass fixed filter ourtput samples
Reconstructed samples right before DBF
Prediction samples
Prediction samples pass fixed filter output samples
Residual samples
Residual samples pass fixed filter output samples
Reconstructed samples right before SAO
Reconstructed samples right before SAO pass fixed filter ourtput samples
Online ALF filter inputs

FIG. 14

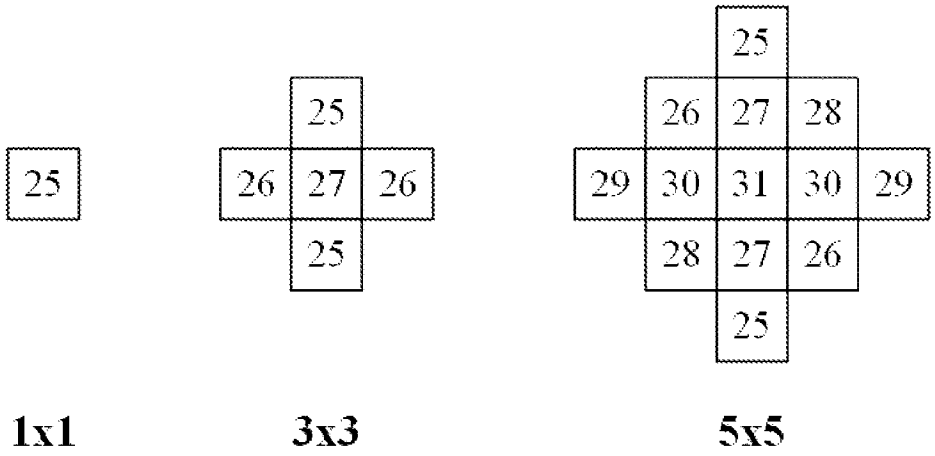

25
25
26 27 26
25
25
26 27 28
29 30 31 30 29
28 27 26
25
1x1
3x3
5x5

FIG. 15

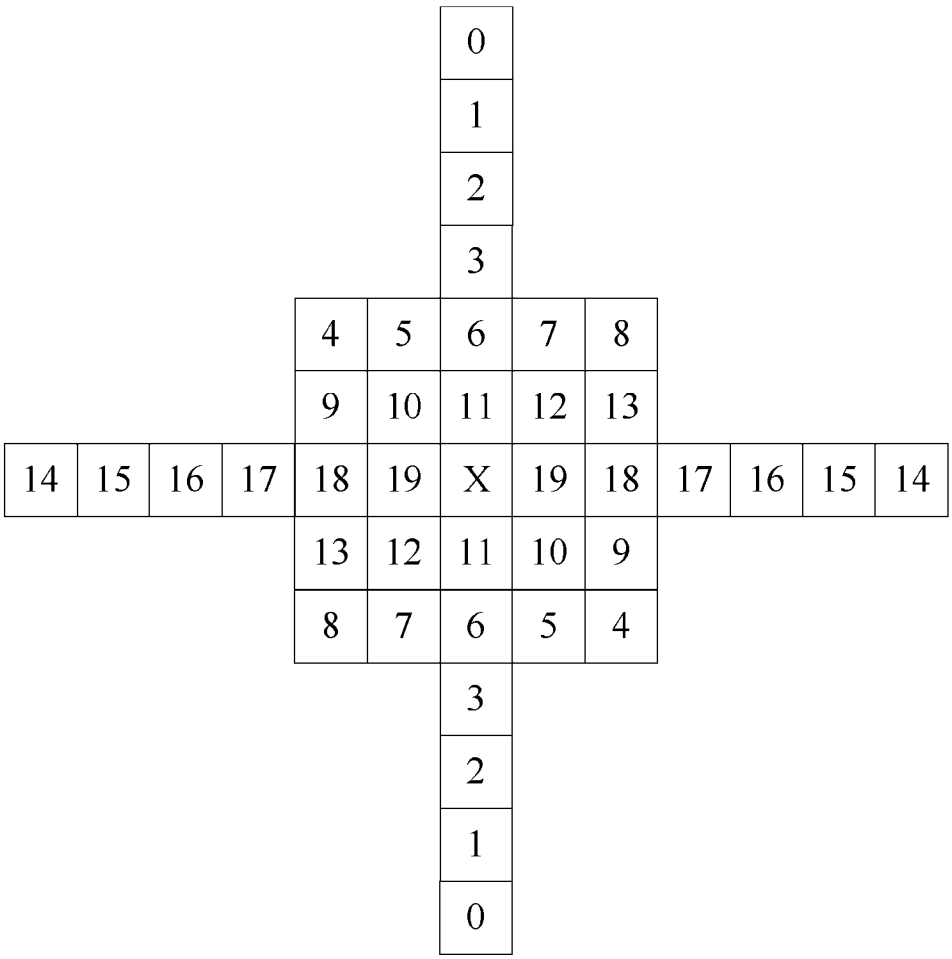
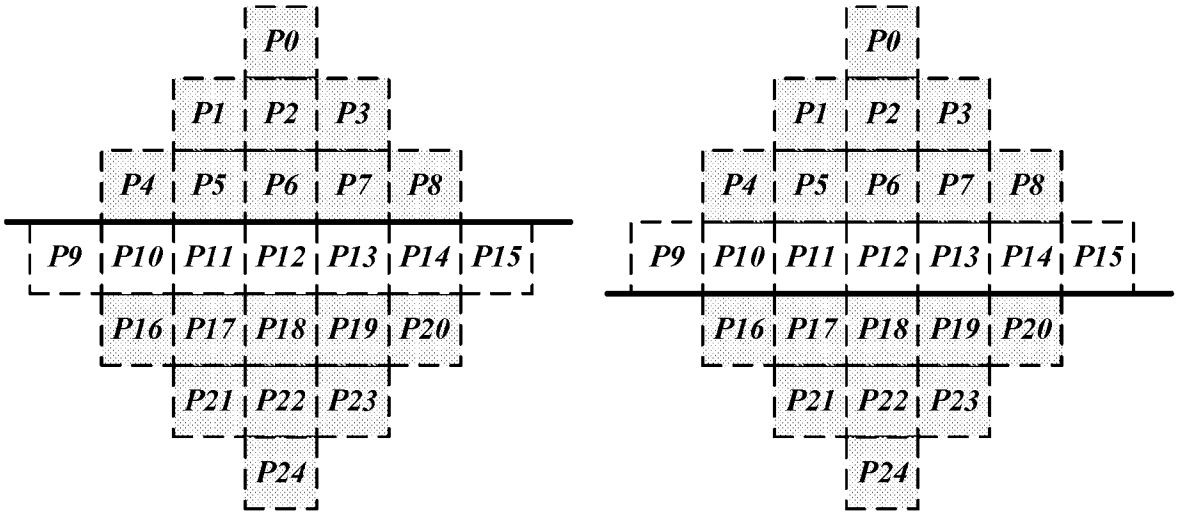

0
1
2
3
4 5 6 7 8
9 10 11 12 13
14 15 16 17 18 19 X 19 18 17 16 15 14
13 12 11 10 9
8 7 6 5 4
3
2
1
0

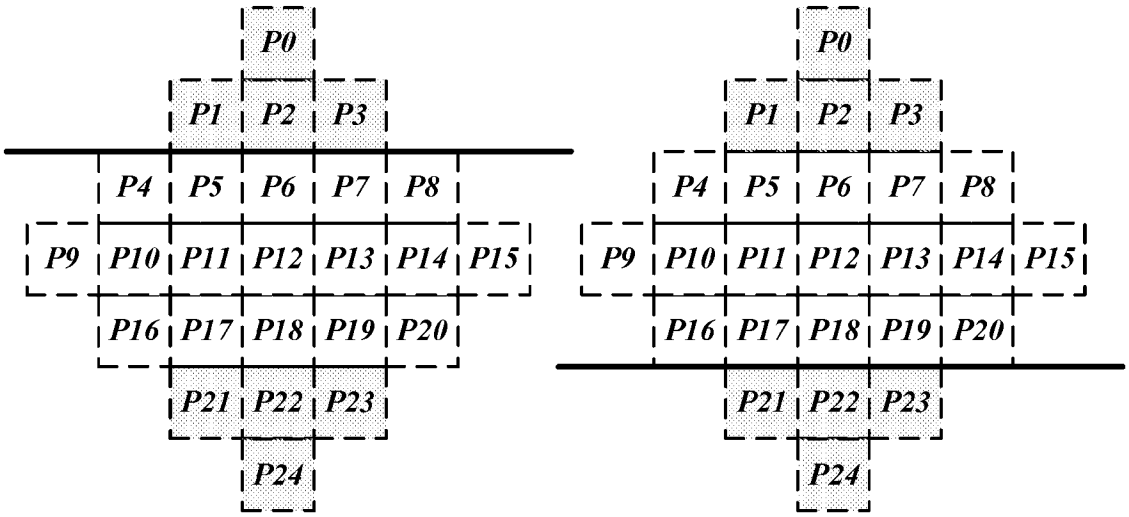
FIG. 16



P'4=P'16=P10
P'1=P'5=P'17=P'21=P11
P'0=P'2=P'6=P12
P'18=P'22=P'24=P12

P'3=P'7=P'19=P'23=P13
P'8=P'20=P14
P'i=Pi , i=9...15

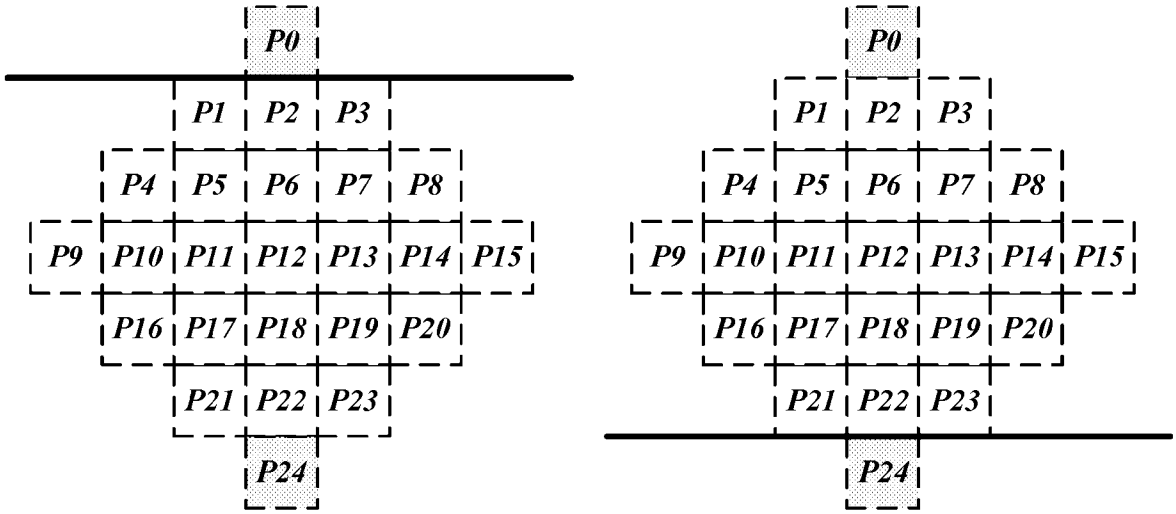
FIG. 17A



P'1=P5 , P'21=P17
P'0=P'2=P6
P'22=P'24=P18

P'3=P7 , P'23=P19
P'i=Pi , i=4...20

FIG. 17B



P'0=P2
P'24=P22
P'i=Pi , i=1...23

FIG. 17C

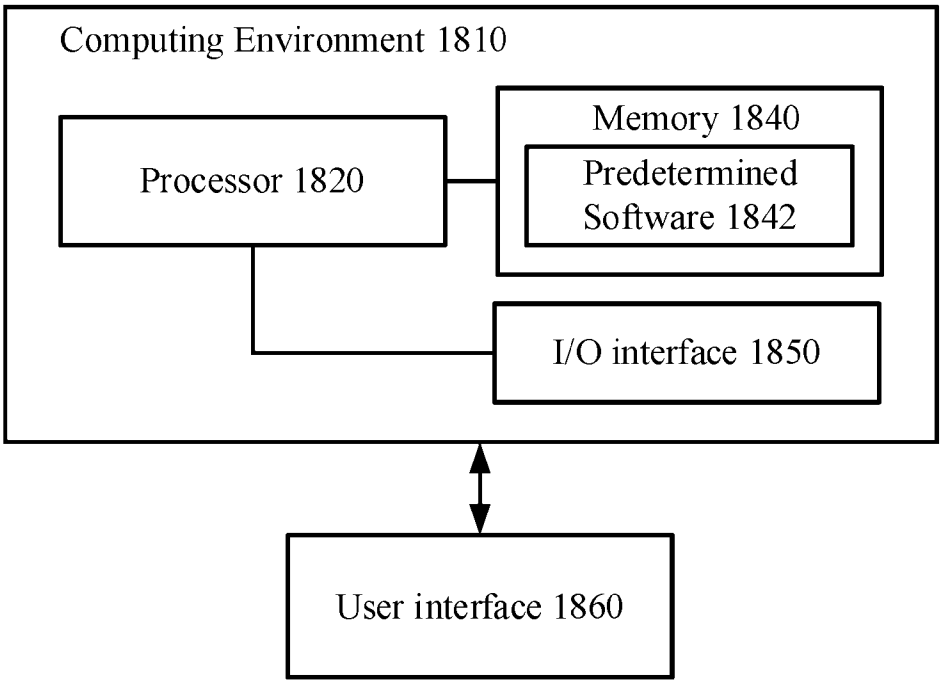


FIG. 18

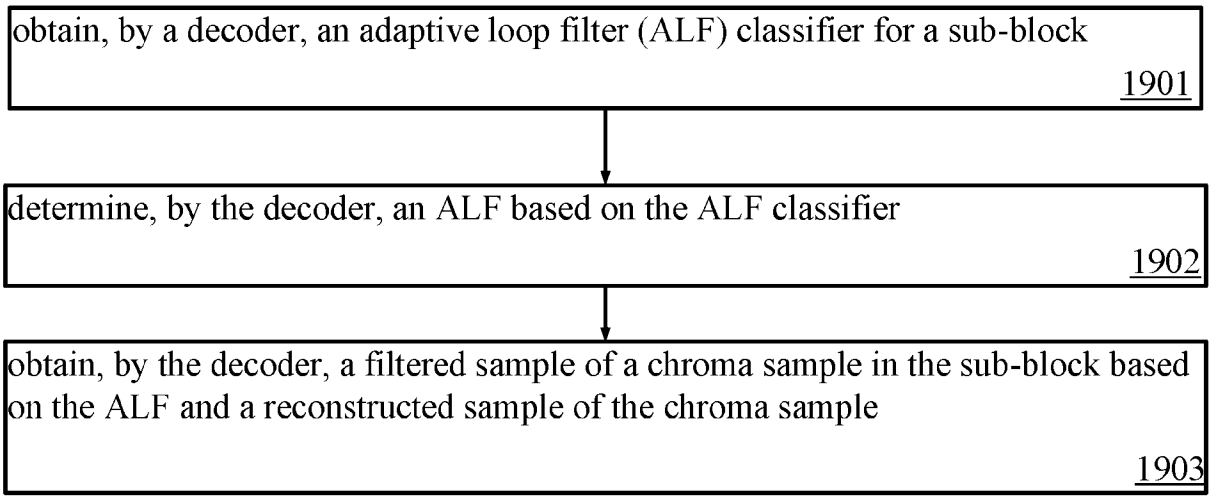

obtain, by a decoder, an adaptive loop filter (ALF) classifier for a sub-block
1901
determine, by the decoder, an ALF based on the ALF classifier
1902
obtain, by the decoder, a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample
1903

FIG. 19

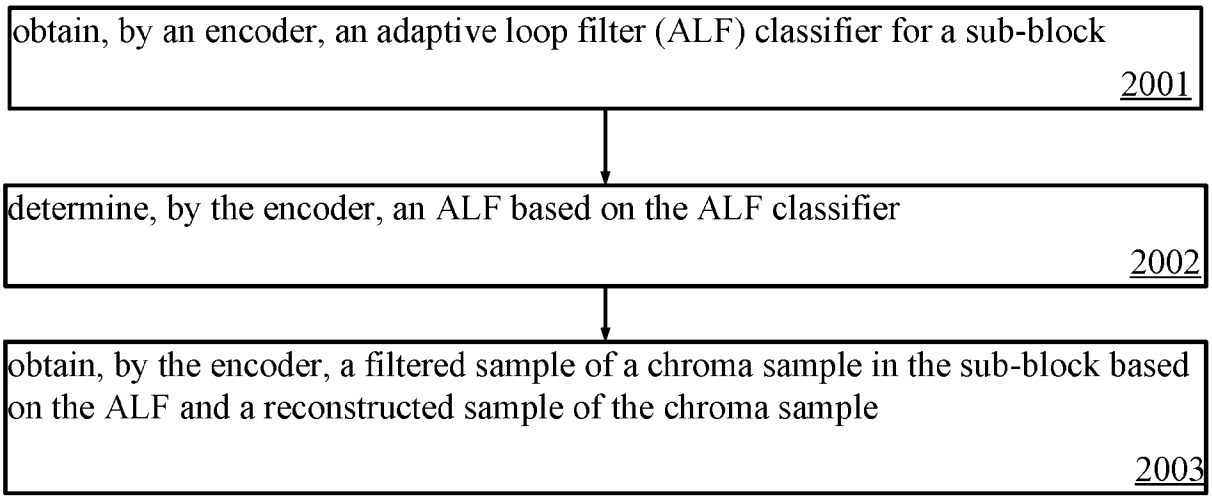

obtain, by an encoder, an adaptive loop filter (ALF) classifier for a sub-block
2001
determine, by the encoder, an ALF based on the ALF classifier
2002
obtain, by the encoder, a filtered sample of a chroma sample in the sub-block based on the ALF and a reconstructed sample of the chroma sample
2003

FIG. 20

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2024/054502

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/117(2014.01)i; **H04N 19/82**(2014.01)i; **H04N 19/132**(2014.01)i; **H04N 19/186**(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 19/117(2014.01); H04N 19/132(2014.01); H04N 19/17(2014.01); H04N 19/176(2014.01); H04N 19/186(2014.01); H04N 19/82(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: decoder, encoder, adaptive loop filter (ALF), classifier, sub-block, chroma, luma

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2021-083257 A1 (BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. et al.) 06 May 2021 (2021-05-06) paragraphs [0004], [0053], [0098], [0105], [0107]-[0108], [0111]; claims 45, 51-52, 57, 67, 78, 81; and figure 4A	1-2,7,12-14,19,24-32
Y		3-6,8-11,15-18,20-23
Y	WO 2023-200966 A1 (BEIJING DAJIA INTERNET INFORMATION TECHNOLOGY CO., LTD.) 19 October 2023 (2023-10-19) paragraphs [0127]-[0129]	3-4,8-11,15-16,20-23
Y	US 2023-0101318 A1 (QUALCOMM INCORPORATED) 30 March 2023 (2023-03-30) paragraphs [0119]-[0120]; and figure 6	5-6,17-18
A	US 2023-0353790 A1 (PANASONIC INTELLECTUAL PROPERTY CORPORATION OF AMERICA) 02 November 2023 (2023-11-02) claims 3-4	1-32

✓ 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
19 February 2025	**19 February 2025**
Name and mailing address of the ISA/KR **Korean Intellectual Property Office** **189 Cheongsa-ro, Seo-gu, Daejeon** **35208, Republic of Korea** Facsimile No. **+82-42-481-8578**	Authorized officer **YANG, Jeong Rok** Telephone No. **+82-42-481-5709**

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2024/054502

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2023-0269371 A1 (LG ELECTRONICS INC.) 24 August 2023 (2023-08-24) claims 16-17	1-32

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/US2024/054502

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
WO	2021-083257	A1	06 May 2021	CN	117528097	A	06 February 2024
				EP	4042692	A1	17 August 2022
				JP	2024-010138	A	23 January 2024
				JP	7485761	B2	16 May 2024
				KR	10-2022-0084060	A	21 June 2022
				KR	10-2717414	B1	15 October 2024
				US	11736697	B2	22 August 2023
				US	2023-0396773	A1	07 December 2023
				WO	2021-083259	A1	06 May 2021
WO	2023-200966	A1	19 October 2023	CN	118923110	A	08 November 2024
				CN	119013979	A	22 November 2024
US	2023-0101318	A1	30 March 2023	CN	117981303	A	03 May 2024
				EP	4409887	A1	07 August 2024
				KR	10-2024-0090151	A	21 June 2024
				WO	2023-056159	A1	06 April 2023
US	2023-0353790	A1	02 November 2023	CN	114270850	A	01 April 2022
				US	11528508	B2	13 December 2022
				US	12028554	B2	02 July 2024
				US	2022-0159307	A1	19 May 2022
				US	2024-0276021	A1	15 August 2024
				WO	2021-025169	A1	11 February 2021
US	2023-0269371	A1	24 August 2023	CN	114424531	A	29 April 2022
				KR	10-2022-0019243	A	16 February 2022
				US	11677939	B2	13 June 2023
				US	12069254	B2	20 August 2024
				US	2022-0132114	A1	28 April 2022
				US	2024-0373001	A1	07 November 2024
				WO	2021-006633	A1	14 January 2021