US 20250267259A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2025/0267259 A1**

CHERNYAK et al. (43) **Pub. Date: Aug. 21, 2025**

(54) **INVERSE PRE-FILTER FOR IMAGE AND VIDEO COMPRESSION**

(71) Applicant: **Tencent America LLC**, Palo Alto, CA (US)

(72) Inventors: **Roman CHERNYAK**, Palo Alto, CA (US); **Lien-Fei Chen**, Palo Alto, CA (US); **Biao Wang**, Palo Alto, CA (US); **Motong Xu**, Palo Alto, CA (US); **Shan Liu**, Palo Alto, CA (US); **Ziyue Xiang**, Palo Alto, CA (US); **Minhao Tang**, Palo Alto, CA (US); **Yonguk Yoon**, Palo Alto, CA (US)

(21) Appl. No.: **19/048,596**

(22) Filed: **Feb. 7, 2025**

Related U.S. Application Data

(60) Provisional application No. 63/555,884, filed on Feb. 20, 2024.

Publication Classification

(51) **Int. Cl.**

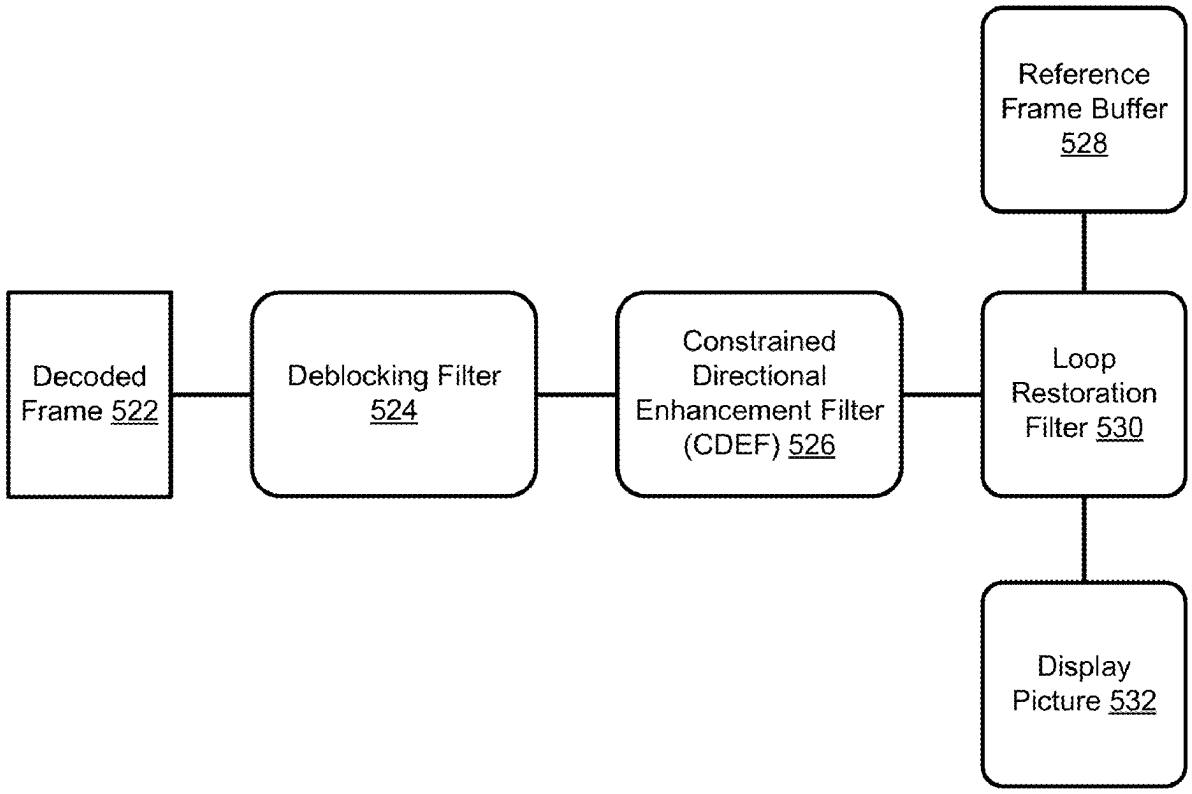
H04N 19/117	(2014.01)
H04N 19/184	(2014.01)
H04N 19/82	(2014.01)

(52) **U.S. Cl.**

CPC ***H04N 19/117*** (2014.11); ***H04N 19/184*** (2014.11); ***H04N 19/82*** (2014.11)

(57) **ABSTRACT**

The various implementations described herein include methods and systems for coding video. In one aspect, a method includes receiving a video bitstream comprising a current picture. The method includes obtaining a filtered current picture by applying at least one in-loop filter to the current picture. The method also includes obtaining a reconstructed current picture by applying an inverse pre-filter to the filtered current picture.



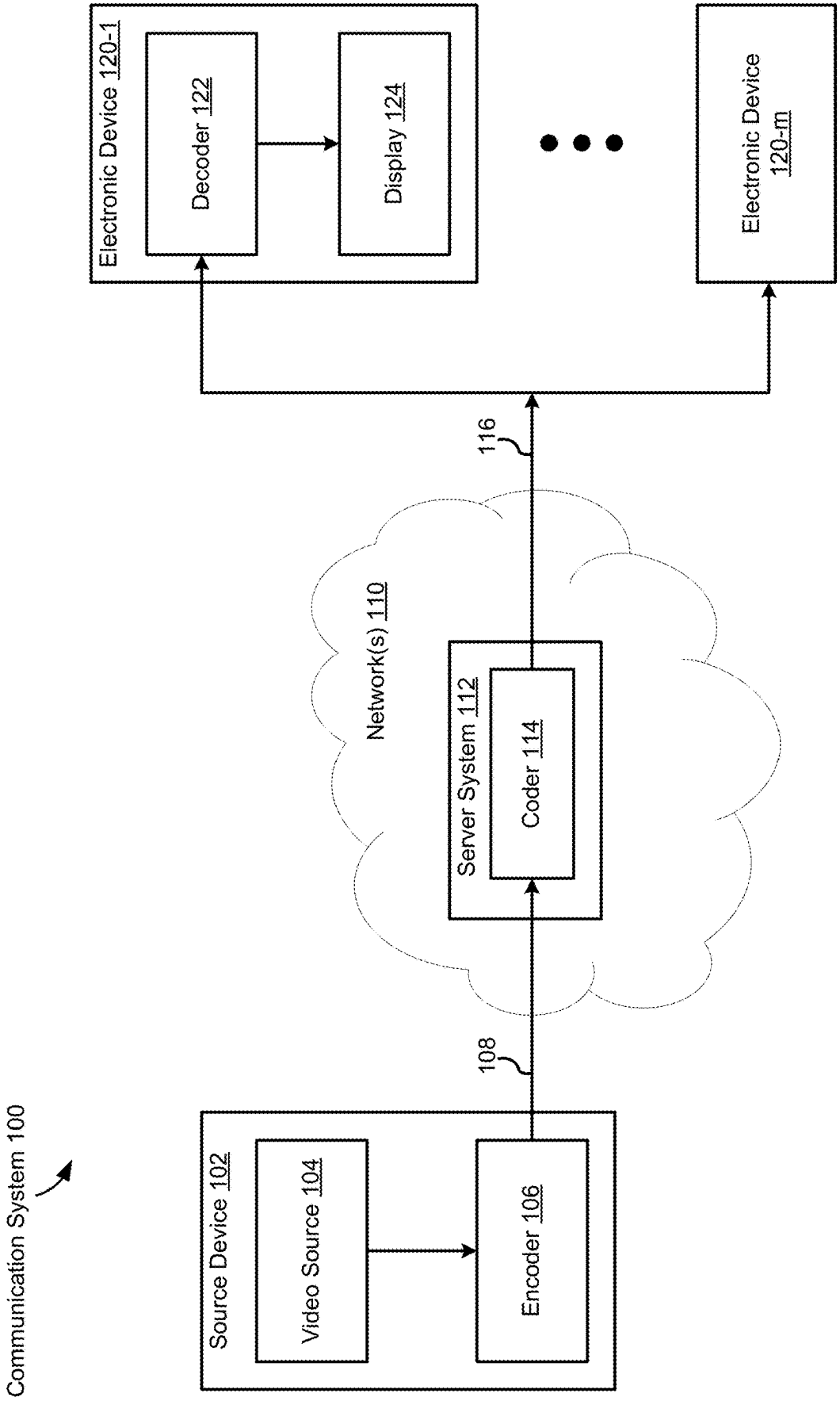

Communication System 100
Source Device 102
Video Source 104
Encoder 106
108
Network(s) 110
Server System 112
Coder 114
116
Electronic Device 120-1
Decoder 122
Display 124
Electronic Device 120-m

FIG. 1

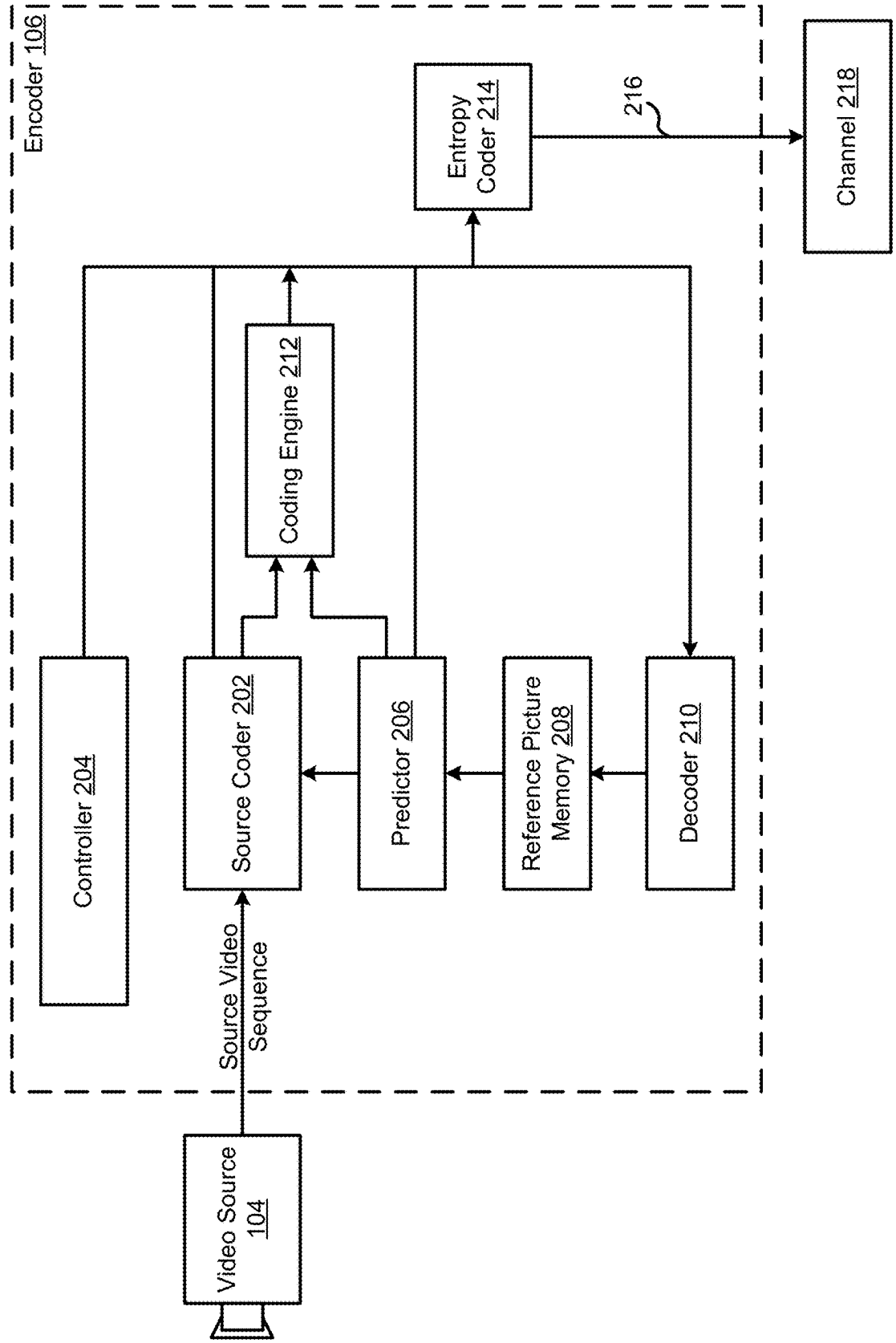

Encoder 106
Entropy Coder 214
216
Channel 218
Coding Engine 212
Controller 204
Source Coder 202
Predictor 206
Reference Picture Memory 208
Decoder 210
Source Video Sequence
Video Source 104

FIG. 2A

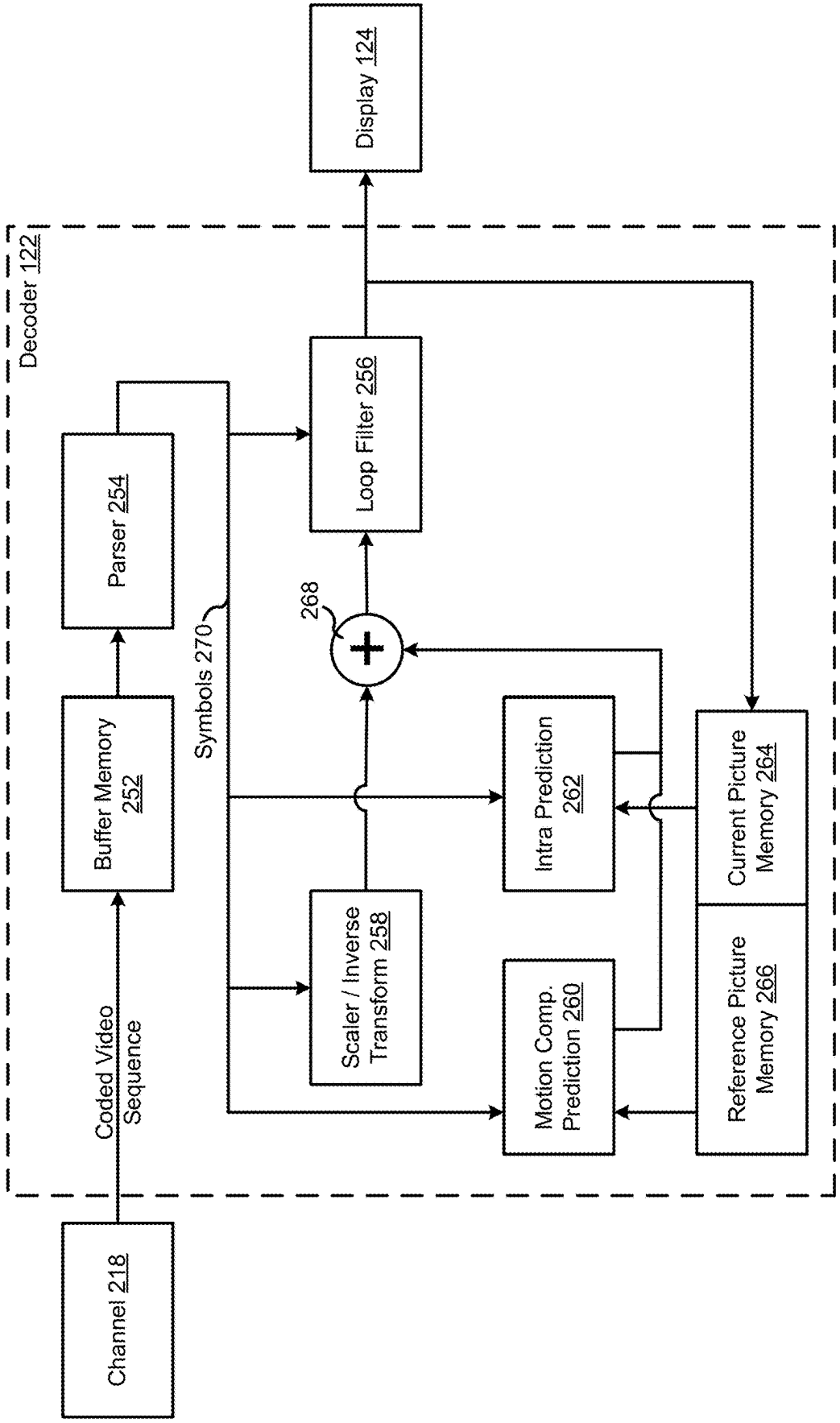

Channel 218
Coded Video Sequence
Decoder 122
Buffer Memory 252
Parser 254
Symbols 270
Scaler / Inverse Transform 258
Motion Comp. Prediction 260
Intra Prediction 262
268
Loop Filter 256
Current Picture Memory 264
Reference Picture Memory 266
Display 124

FIG. 2B

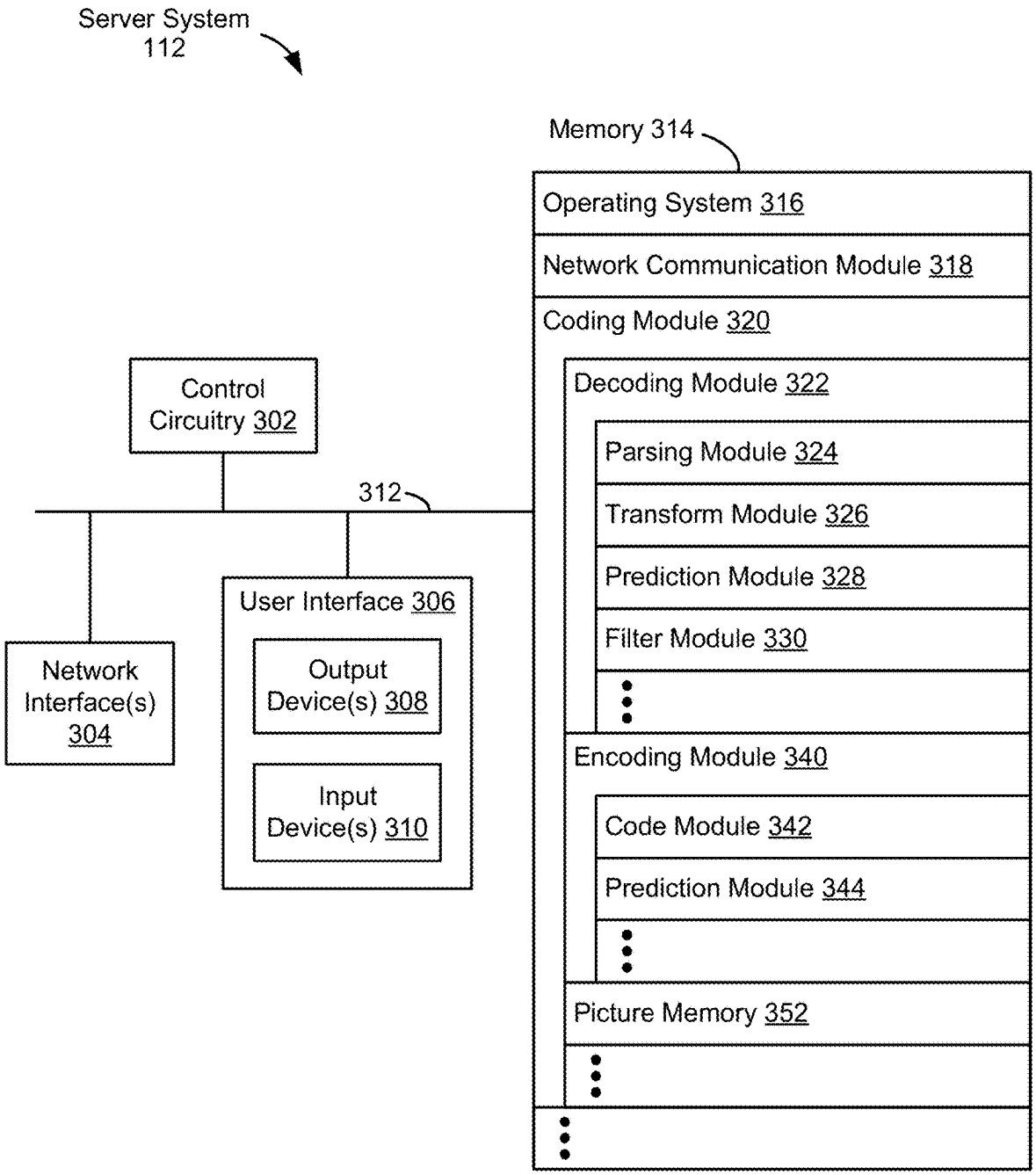

Server System
112
Memory 314
Operating System 316
Network Communication Module 318
Coding Module 320
Decoding Module 322
Parsing Module 324
Transform Module 326
Prediction Module 328
Filter Module 330
Encoding Module 340
Code Module 342
Prediction Module 344
Picture Memory 352
Control
Circuitry 302
312
User Interface 306
Output
Device(s) 308
Input
Device(s) 310
Network
Interface(s)
304

FIG. 3

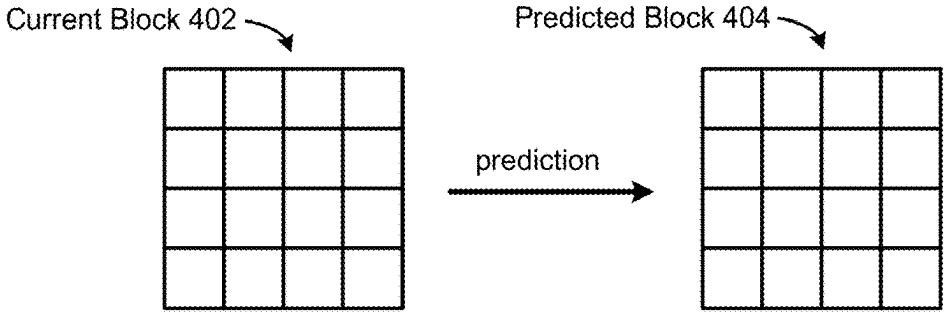


FIG. 4A

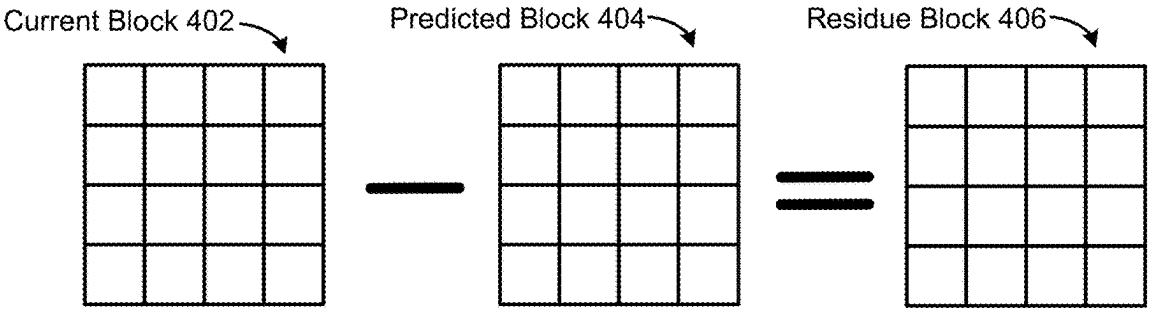


FIG. 4B

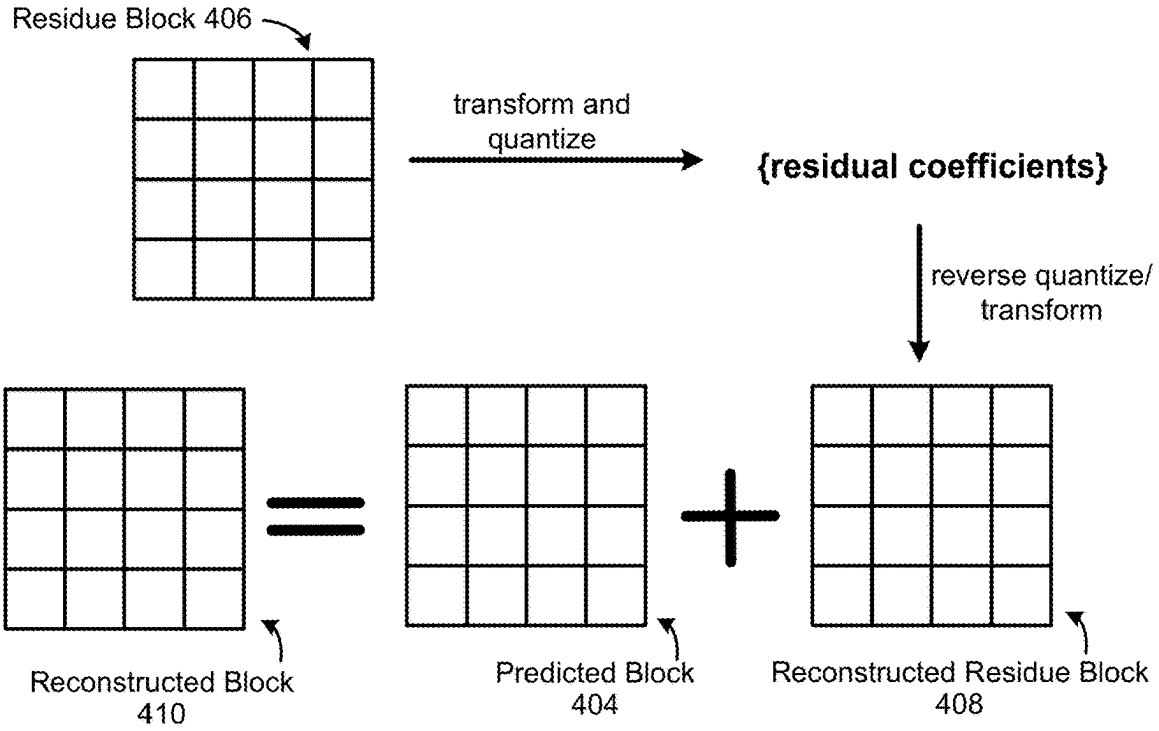


FIG. 4C

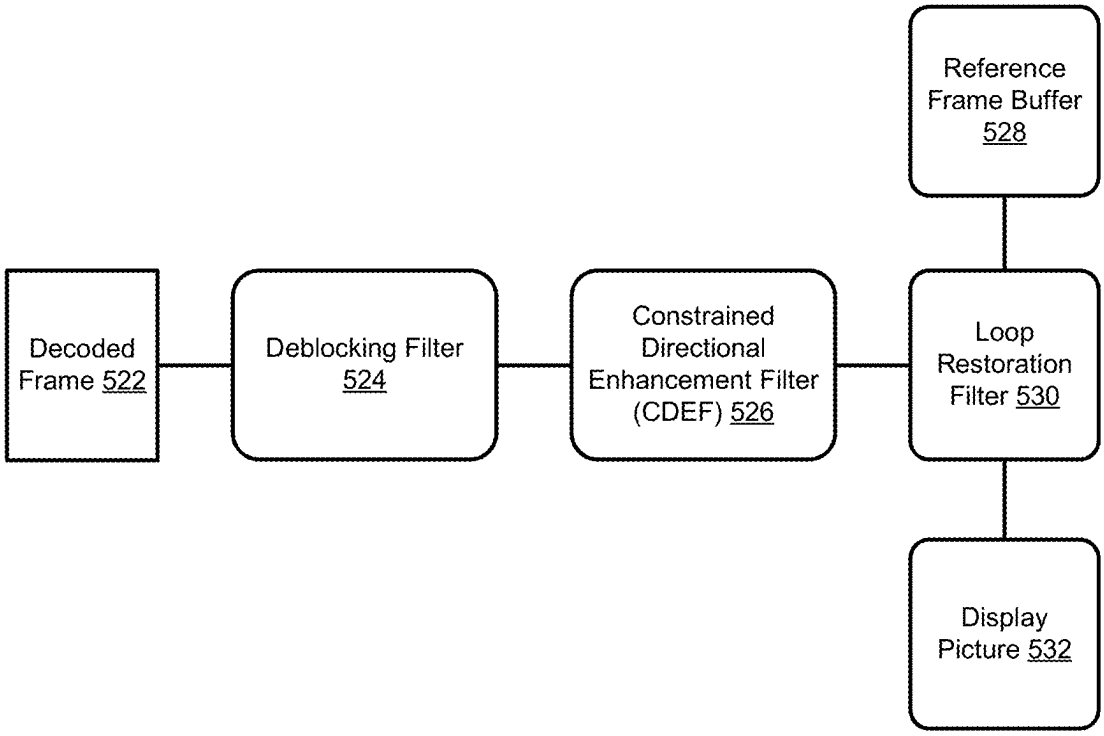

Reference Frame Buffer 528
Decoded Frame 522
Deblocking Filter 524
Constrained Directional Enhancement Filter (CDEF) 526
Loop Restoration Filter 530
Display Picture 532

FIG. 5

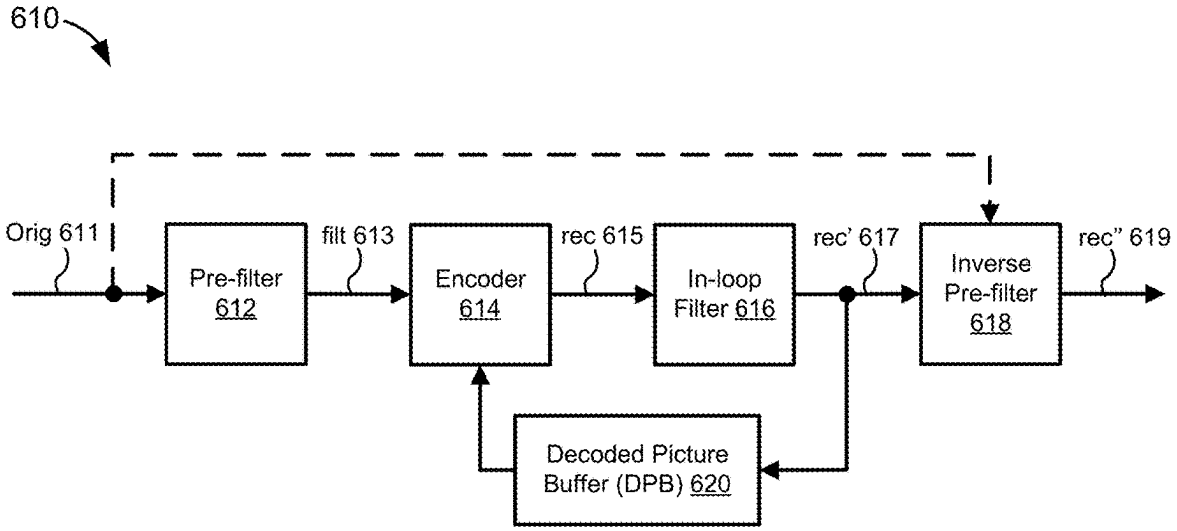

610
Orig 611
Pre-filter 612
filt 613
Encoder 614
rec 615
In-loop Filter 616
rec' 617
Inverse Pre-filter 618
rec" 619
Decoded Picture Buffer (DPB) 620

FIG. 6A

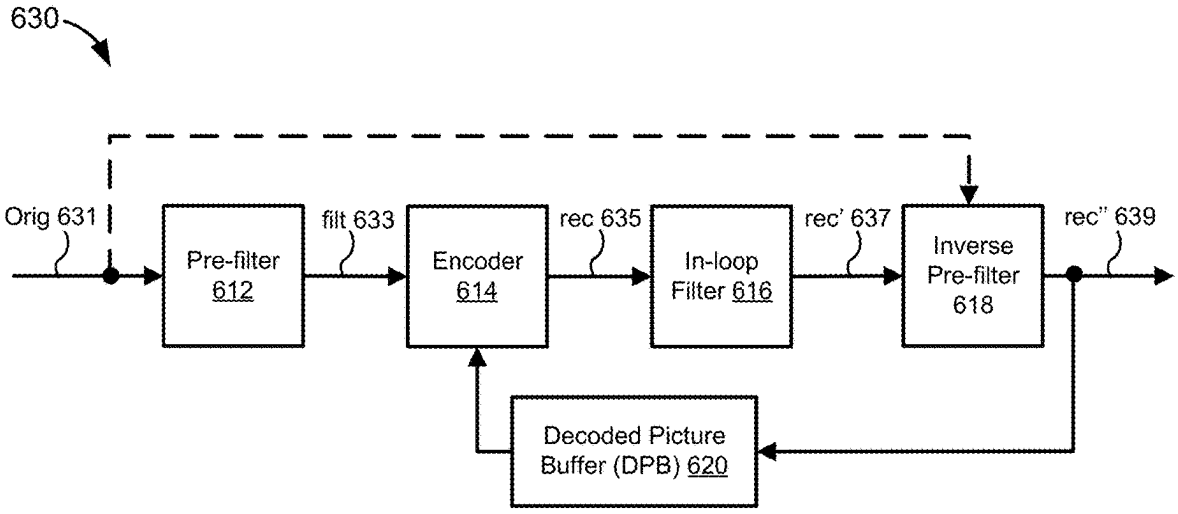

630
Orig 631
Pre-filter 612
filt 633
Encoder 614
rec 635
In-loop Filter 616
rec' 637
Inverse Pre-filter 618
rec" 639
Decoded Picture Buffer (DPB) 620

FIG. 6B

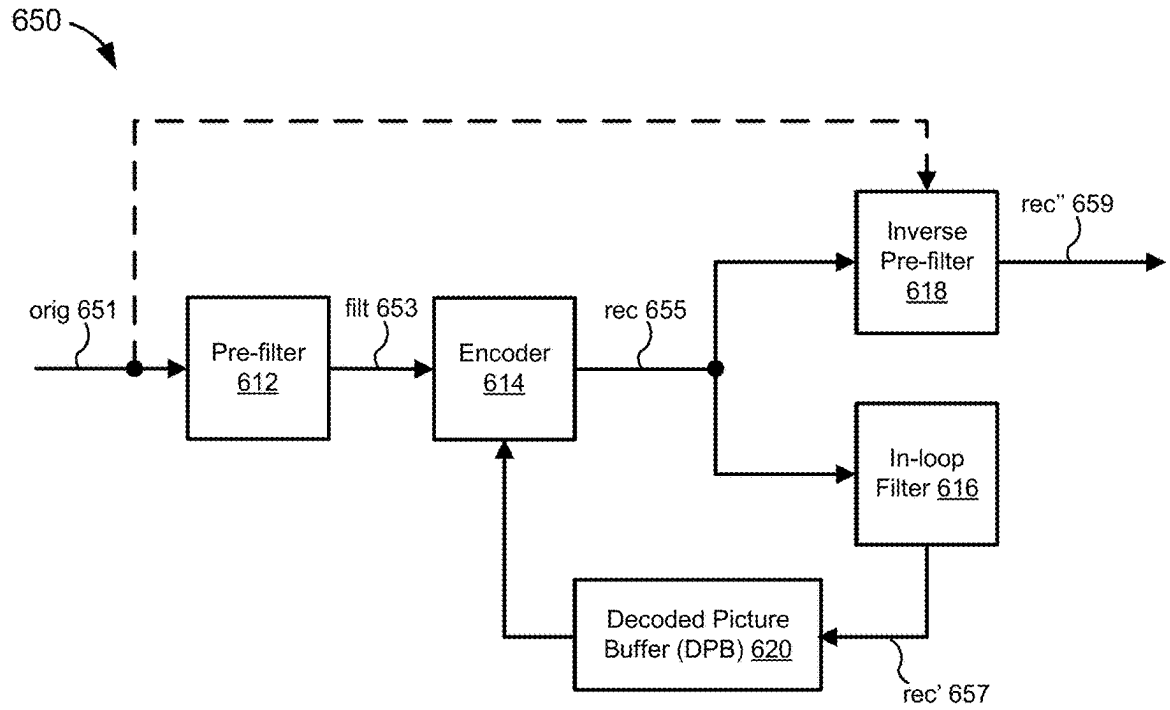

650
orig 651
Pre-filter 612
filt 653
Encoder 614
rec 655
Inverse Pre-filter 618
rec" 659
In-loop Filter 616
Decoded Picture Buffer (DPB) 620
rec' 657

FIG. 6C

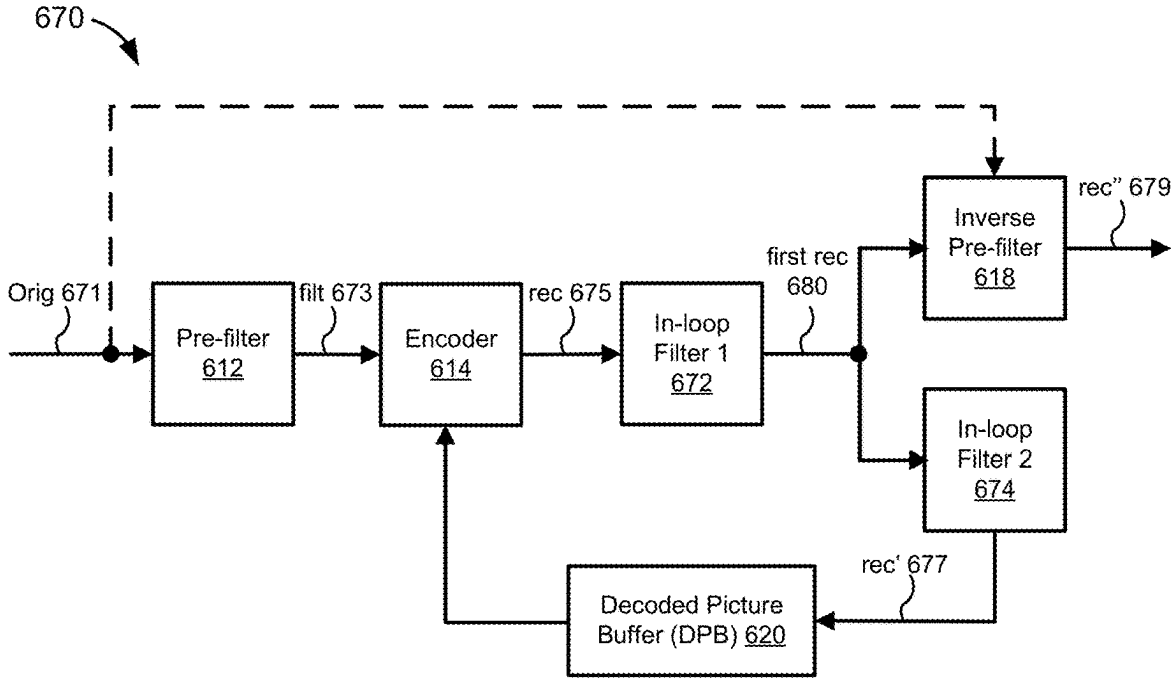

670
Orig 671
Pre-filter 612
filt 673
Encoder 614
rec 675
In-loop Filter 1 672
first rec 680
Inverse Pre-filter 618
rec" 679
In-loop Filter 2 674
rec' 677
Decoded Picture Buffer (DPB) 620

FIG. 6D

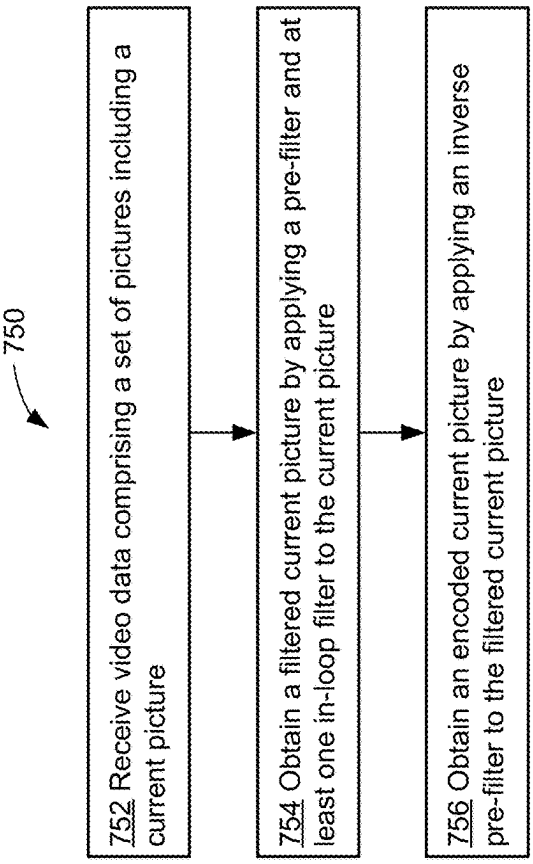

750
752 Receive video data comprising a set of pictures including a current picture
754 Obtain a filtered current picture by applying a pre-filter and at least one in-loop filter to the current picture
756 Obtain an encoded current picture by applying an inverse pre-filter to the filtered current picture

FIG. 7B

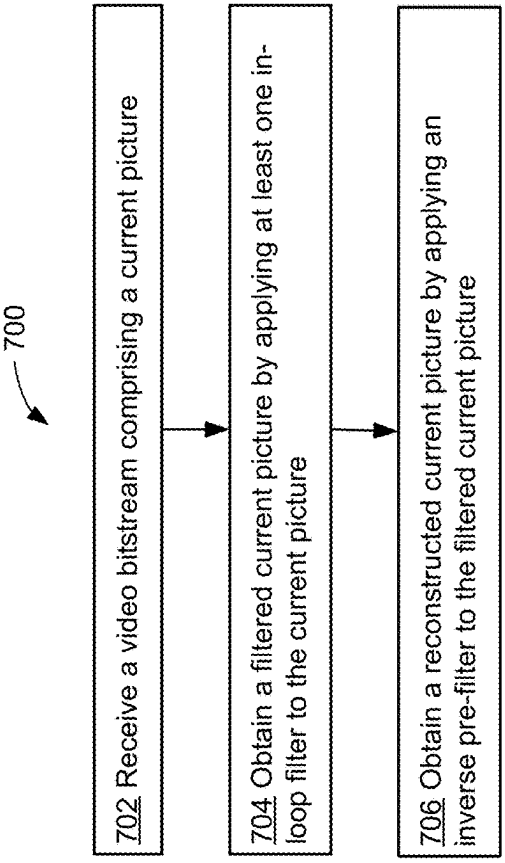

700
702 Receive a video bitstream comprising a current picture
704 Obtain a filtered current picture by applying at least one in-loop filter to the current picture
706 Obtain a reconstructed current picture by applying an inverse pre-filter to the filtered current picture

FIG. 7A

INVERSE PRE-FILTER FOR IMAGE AND VIDEO COMPRESSION

RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Patent Application No. 63/555,884, entitled "Inverse Pre-Filter for Image and Video Compression," filed Feb. 20, 2024, which is hereby incorporated by reference in its entirety.

TECHNICAL FIELD

[0002] The disclosed embodiments relate generally to video coding, including but not limited to systems and methods for applying inverse pre-filtering for image and video compression.

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. The video coding can be performed by hardware and/or software on an electronic/client device or a server providing a cloud service.

[0004] 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. Multiple video codec standards have been developed. For example, High-Efficiency Video Coding (HEVC/H.265) is a video compression standard designed as part of the MPEG-H project. ITU-T and ISO/IEC published the HEVC/H.265 standard in 2013 (version 1), 2014 (version 2), 2015 (version 3), and 2016 (version 4). Versatile Video Coding (VVC/H.266) is a video compression standard intended as a successor to HEVC. ITU-T and ISO/IEC published the VVC/H.266 standard in 2020 (version 1) and 2022 (version 2). AOMedia Video 1 (AV1) is an open video coding format designed as an alternative to HEVC. On Jan. 8, 2019, a validated version 1.0.0 with Errata 1 of the specification was released. Enhanced Compression Model (ECM) is a video coding standard that is currently under development. ECM aims to significantly improve compression efficiency beyond existing standards like HEVC/H.265 and VVC, essentially allowing for higher quality video at lower bitrates.

SUMMARY

[0005] The present disclosure describes amongst other things, a set of methods for video (image) compression, more specifically related to applying filtering to increase video compression performance. Many existing video and image codecs use a pre-filtering step prior to compressing video data, e.g., to make the video signals easier to compress and improve compression efficiency. However, such approaches can cause reconstruction distortion since the pre-filters unavoidably remove some part of useful information that may not be reconstructed at the decoder side. The present disclosure describes implementing inverse pre-filters to recover information that was lost in pre-filtering step thereby improving coding accuracy. For example, an inverse pre-filtering step may be added to an encoding/decoding process (e.g., during or after a loop filtering process) to recover some information lost during a pre-filtering step.

[0006] In accordance with some embodiments, a method of video decoding is provided. The method includes (i) receiving a video bitstream (e.g., a coded video sequence) comprising a current picture; (ii) obtaining a filtered current picture by applying a pre-filter and at least one in-loop filter to the current picture; and (iii) obtaining a reconstructed current picture by applying an inverse pre-filter to the filtered current picture.

[0007] In accordance with some embodiments, a method of video encoding is provided. The method includes (i) receiving video data (e.g., a source video sequence) comprising a set of pictures including a current picture; (ii) obtaining a filtered current picture by applying a pre-filter and at least one in-loop filter to the current picture; and (iii) obtaining an encoded current picture by applying an inverse pre-filter to the filtered current picture.

[0008] In accordance with some embodiments, a method of processing visual media data includes: (i) obtaining a source video sequence that comprises a plurality of frames; and (ii) performing a conversion between the source video sequence and a video bitstream of visual media data according to a format rule, where (a) the video bitstream comprises a current picture; and (b) the format rule specifies that (1) a filtered current picture is to be obtained by applying at least one in-loop filter to the current picture; and (2) a reconstructed current picture is to be obtained by applying an inverse pre-filter to the filtered current picture.

[0009] In accordance with some embodiments, a method of video decoding is provided. The method includes (i) receiving a video bitstream (e.g., a coded video sequence) comprising a current picture; (ii) obtaining a filtered current picture by applying a pre-filter and at least one in-loop filter to the current picture; (iii) storing the filtered current picture in a decoded picture buffer for use in in-loop filtering; and (iv) obtaining a reconstructed current picture by applying an inverse pre-filter to the current picture.

[0010] In accordance with some embodiments, a computing system is provided, such as a streaming system, a server system, a personal computer system, or other electronic device. The computing system includes control circuitry and memory storing one or more sets of instructions. The one or more sets of instructions including instructions for performing any of the methods described herein. In some embodiments, the computing system includes an encoder component and a decoder component (e.g., a transcoder).

[0011] In accordance with some embodiments, a non-transitory computer-readable storage medium is provided. The non-transitory computer-readable storage medium stores one or more sets of instructions for execution by a computing system. The one or more sets of instructions including instructions for performing any of the methods described herein.

[0012] Thus, devices and systems are disclosed with methods for encoding and decoding video. Such methods,

devices, and systems may complement or replace conventional methods, devices, and systems for video encoding/decoding.

[0013] The features and advantages described in the specification are not necessarily all-inclusive and, in particular, some additional features and advantages will be apparent to one of ordinary skill in the art in view of the drawings, specification, and claims provided in this disclosure. Moreover, it should be noted that the language used in the specification has been principally selected for readability and instructional purposes and has not necessarily been selected to delineate or circumscribe the subject matter described herein.

BRIEF DESCRIPTION OF THE DRAWINGS

[0014] So that the present disclosure can be understood in greater detail, a more particular description can be had by reference to the features of various embodiments, some of which are illustrated in the appended drawings. The appended drawings, however, merely illustrate pertinent features of the present disclosure and are therefore not necessarily to be considered limiting, for the description can admit to other effective features as the person of skill in this art will appreciate upon reading this disclosure.

[0015] FIG. **1** is a block diagram illustrating an example communication system in accordance with some embodiments.

[0016] FIG. **2**A is a block diagram illustrating example elements of an encoder component in accordance with some embodiments.

[0017] FIG. **2**B is a block diagram illustrating example elements of a decoder component in accordance with some embodiments.

[0018] FIG. **3** is a block diagram illustrating an example server system in accordance with some embodiments.

[0019] FIGS. **4**A-**4**C illustrate example prediction blocks, residue blocks, and reconstructed blocks according to some embodiments.

[0020] FIG. **5** illustrates example in-loop filtering stages according to some embodiments.

[0021] FIGS. **6**A-**6**D illustrate example configurations for implementing an inverse pre-filter on the encoder side, in accordance with some embodiments.

[0022] FIG. **7**A illustrates an example video decoding process in accordance with some embodiments.

[0023] FIG. **7**B illustrates an example video encoding process in accordance with some embodiments.

[0024] In accordance with common practice, the various features illustrated in the drawings are not necessarily drawn to scale, and like reference numerals can be used to denote like features throughout the specification and figures.

DETAILED DESCRIPTION

[0025] The present disclosure describes video/image compression techniques including applying inverse prefiltering to filtered pictures. Existing video and image codecs may use pre-filtering to remove some noise prior to encoding. However, the pre-filters also remove some useful information from the filtered pictures, which degrades coding accuracy. Some embodiments disclosed herein include obtaining a filtered current picture by applying a pre-prefilter and at least one in-loop filter to the current picture then obtaining a reconstructed current picture by applying an inverse pre-filter to the filtered current picture. In this way, coding accuracy may be improved by adding back some information that was lost due to the pre-filtering. By performing the inverse pre-filtering after the in-loop filter(s), the in-loop filtering accuracy can be improved (by operating on pictures without the noise removed by the pre-filter). Thus, performing the pre-filtering operation prior to any in-loop filtering can improve the performance of the in-loop filtering, while performing the inverse pre-filtering operation after the in-lop filtering restores some of the information lost in the pre-filtering operation.

Example Systems and Devices

[0026] FIG. **1** is a block diagram illustrating a communication system **100** in accordance with some embodiments. The communication system **100** includes a source device **102** and a plurality of electronic devices **120** (e.g., electronic device **120-1** to electronic device **120-***m*) that are communicatively coupled to one another via one or more networks. In some embodiments, the communication system **100** is a streaming system, e.g., for use with video-enabled applications such as video conferencing applications, digital TV applications, and media storage and/or distribution applications.

[0027] The source device **102** includes a video source **104** (e.g., a camera component or media storage) and an encoder component **106**. In some embodiments, the video source **104** is a digital camera (e.g., configured to create an uncompressed video sample stream). The encoder component **106** generates one or more encoded video bitstreams from the video stream. The video stream from the video source **104** may be high data volume as compared to the encoded video bitstream **108** generated by the encoder component **106**. Because the encoded video bitstream **108** is lower data volume (less data) as compared to the video stream from the video source, the encoded video bitstream **108** requires less bandwidth to transmit and less storage space to store as compared to the video stream from the video source **104**. In some embodiments, the source device **102** does not include the encoder component **106** (e.g., is configured to transmit uncompressed video to the network(s) **110**).

[0028] The one or more networks **110** represents any number of networks that convey information between the source device **102**, the server system **112**, and/or the electronic devices **120**, including for example wireline (wired) and/or wireless communication networks. The one or more networks **110** may exchange data in circuit-switched and/or packet-switched channels. Representative networks include telecommunications networks, local area networks, wide area networks and/or the Internet.

[0029] The one or more networks **110** include a server system **112** (e.g., a distributed/cloud computing system). In some embodiments, the server system **112** is, or includes, a streaming server (e.g., configured to store and/or distribute video content such as the encoded video stream from the source device **102**). The server system **112** includes a coder component **114** (e.g., configured to encode and/or decode video data). In some embodiments, the coder component **114** includes an encoder component and/or a decoder component. In various embodiments, the coder component **114** is instantiated as hardware, software, or a combination thereof. In some embodiments, the coder component **114** is configured to decode the encoded video bitstream **108** and re-encode the video data using a different encoding standard

and/or methodology to generate encoded video data **116**. In some embodiments, the server system **112** is configured to generate multiple video formats and/or encodings from the encoded video bitstream **108**. In some embodiments, the server system **112** functions as a Media-Aware Network Element (MANE). For example, the server system **112** may be configured to prune the encoded video bitstream **108** for tailoring potentially different bitstreams to one or more of the electronic devices **120**. In some embodiments, a MANE is provided separate from the server system **112**.

[0030] The electronic device **120-1** includes a decoder component **122** and a display **124**. In some embodiments, the decoder component **122** is configured to decode the encoded video data **116** to generate an outgoing video stream that can be rendered on a display or other type of rendering device. In some embodiments, one or more of the electronic devices **120** does not include a display component (e.g., is communicatively coupled to an external display device and/or includes a media storage). In some embodiments, the electronic devices **120** are streaming clients. In some embodiments, the electronic devices **120** are configured to access the server system **112** to obtain the encoded video data **116**.

[0031] The source device and/or the plurality of electronic devices **120** are sometimes referred to as "terminal devices" or "user devices." In some embodiments, the source device **102** and/or one or more of the electronic devices **120** are instances of a server system, a personal computer, a portable device (e.g., a smartphone, tablet, or laptop), a wearable device, a video conferencing device, and/or other type of electronic device.

[0032] In example operation of the communication system **100**, the source device **102** transmits the encoded video bitstream **108** to the server system **112**. For example, the source device **102** may code a stream of pictures that are captured by the source device. The server system **112** receives the encoded video bitstream **108** and may decode and/or encode the encoded video bitstream **108** using the coder component **114**. For example, the server system **112** may apply an encoding to the video data that is more optimal for network transmission and/or storage. The server system **112** may transmit the encoded video data **116** (e.g., one or more coded video bitstreams) to one or more of the electronic devices **120**. Each electronic device **120** may decode the encoded video data **116** and optionally display the video pictures.

[0033] FIG. **2**A is a block diagram illustrating example elements of the encoder component **106** in accordance with some embodiments. The encoder component **106** receives video data (e.g., a source video sequence) from the video source **104**. In some embodiments, the encoder component includes a receiver (e.g., a transceiver) component configured to receive the source video sequence. In some embodiments, the encoder component **106** receives a video sequence from a remote video source (e.g., a video source that is a component of a different device than the encoder component **106**). The video source **104** may provide the source video sequence in the form of a digital video sample stream that can be of any suitable bit depth (e.g., 8-bit, 10-bit, or 12-bit), any colorspace (e.g., BT.601 Y CrCB, or RGB), and any suitable sampling structure (e.g., Y CrCb 4:2:0 or Y CrCb 4:4:4). In some embodiments, the video source **104** is a storage device storing previously captured/prepared video. In some embodiments, the video source **104** is camera that captures local image information as a video sequence. Video data may be provided as a plurality of individual pictures that impart motion when viewed in sequence. The pictures themselves may be organized as a spatial array of pixels, where each pixel can include one or more samples depending on the sampling structure, color space, etc. in use. A person of ordinary skill in the art can readily understand the relationship between pixels and samples.

[0034] The encoder component **106** is configured to code and/or compress the pictures of the source video sequence into a coded video sequence **216** in real-time or under other time constraints as required by the application. In some embodiments, the encoder component **106** is configured to perform a conversion between the source video sequence and a bitstream of visual media data (e.g., a video bitstream). Enforcing appropriate coding speed is one function of a controller **204**. In some embodiments, the controller **204** controls other functional units as described below and is functionally coupled to the other functional units. Parameters set by the controller **204** may include rate-control-related parameters (e.g., picture skip, quantizer, and/or lambda value of rate-distortion optimization techniques), picture size, group of pictures (GOP) layout, maximum motion vector search range, and so forth. A person of ordinary skill in the art can readily identify other functions of controller **204** as they may pertain to the encoder component **106** being optimized for a certain system design.

[0035] In some embodiments, the encoder component **106** is configured to operate in a coding loop. In a simplified example, the coding loop includes a source coder **202** (e.g., responsible for creating symbols, such as a symbol stream, based on an input picture to be coded and reference picture (s)), and a (local) decoder **210**. The decoder **210** reconstructs the symbols to create the sample data in a similar manner as a (remote) decoder (when compression between symbols and coded video bitstream is lossless). The reconstructed sample stream (sample data) is input to the reference picture memory **208**. As the decoding of a symbol stream leads to bit-exact results independent of decoder location (local or remote), the content in the reference picture memory **208** is also bit exact between the local encoder and remote encoder. In this way, the prediction part of an encoder interprets as reference picture samples the same sample values as a decoder would interpret when using prediction during decoding.

[0036] The operation of the decoder **210** can be the same as of a remote decoder, such as the decoder component **122**, which is described in detail below in conjunction with FIG. **2**B. Briefly referring to FIG. **2**B, however, as symbols are available and encoding/decoding of symbols to a coded video sequence by an entropy coder **214** and the parser **254** can be lossless, the entropy decoding parts of the decoder component **122**, including the buffer memory **252** and the parser **254** may not be fully implemented in the local decoder **210**.

[0037] The decoder technology described herein, except the parsing/entropy decoding, may be to be present, in substantially identical functional form, in a corresponding encoder. For this reason, the disclosed subject matter focuses on decoder operation. Additionally, the description of encoder technologies can be abbreviated as they may be the inverse of the decoder technologies.

[0038] As part of its operation, the source coder **202** may perform motion compensated predictive coding, which codes an input frame predictively with reference to one or more previously-coded frames from the video sequence that were designated as reference frames. In this manner, the coding engine **212** codes differences between pixel blocks of an input frame and pixel blocks of reference frame(s) that may be selected as prediction reference(s) to the input frame. The controller **204** may manage coding operations of the source coder **202**, including, for example, setting of parameters and subgroup parameters used for encoding the video data.

[0039] The decoder **210** decodes coded video data of frames that may be designated as reference frames, based on symbols created by the source coder **202**. Operations of the coding engine **212** may advantageously be lossy processes. When the coded video data is decoded at a video decoder (not shown in FIG. **2**A), the reconstructed video sequence may be a replica of the source video sequence with some errors. The decoder **210** replicates decoding processes that may be performed by a remote video decoder on reference frames and may cause reconstructed reference frames to be stored in the reference picture memory **208**. In this manner, the encoder component **106** stores copies of reconstructed reference frames locally that have common content as the reconstructed reference frames that will be obtained by a remote video decoder (absent transmission errors).

[0040] The predictor **206** may perform prediction searches for the coding engine **212**. That is, for a new frame to be coded, the predictor **206** may search the reference picture memory **208** for sample data (as candidate reference pixel blocks) or certain metadata such as reference picture motion vectors, block shapes, and so on, that may serve as an appropriate prediction reference for the new pictures. The predictor **206** may operate on a sample block-by-pixel block basis to find appropriate prediction references. As determined by search results obtained by the predictor **206**, an input picture may have prediction references drawn from multiple reference pictures stored in the reference picture memory **208**.

[0041] Output of all aforementioned functional units may be subjected to entropy coding in the entropy coder **214**. The entropy coder **214** translates the symbols as generated by the various functional units into a coded video sequence, by losslessly compressing the symbols according to technologies known to a person of ordinary skill in the art (e.g., Huffman coding, variable length coding, and/or arithmetic coding).

[0042] In some embodiments, an output of the entropy coder **214** is coupled to a transmitter. The transmitter may be configured to buffer the coded video sequence(s) as created by the entropy coder **214** to prepare them for transmission via a communication channel **218**, which may be a hardware/software link to a storage device which would store the encoded video data. The transmitter may be configured to merge coded video data from the source coder **202** with other data to be transmitted, for example, coded audio data and/or ancillary data streams (sources not shown). In some embodiments, the transmitter may transmit additional data with the encoded video. The source coder **202** may include such data as part of the coded video sequence. Additional data may comprise temporal/spatial/SNR enhancement layers, other forms of redundant data such as redundant pictures and slices, Supplementary Enhancement Information (SEI) messages, Visual Usability Information (VUI) parameter set fragments, and the like.

[0043] The controller **204** may manage operation of the encoder component **106**. During coding, the controller **204** may assign to each coded picture a certain coded picture type, which may affect the coding techniques that are applied to the respective picture. For example, pictures may be assigned as an Intra Picture (I picture), a Predictive Picture (P picture), or a Bi-directionally Predictive Picture (B Picture). An Intra Picture may be coded and decoded without using any other frame in the sequence as a source of prediction. Some video codecs allow for different types of Intra pictures, including, for example Independent Decoder Refresh (IDR) Pictures. A person of ordinary skill in the art is aware of those variants of I pictures and their respective applications and features, and therefore they are not repeated here. A Predictive picture may be coded and decoded using intra prediction or inter prediction using at most one motion vector and reference index to predict the sample values of each block. A Bi-directionally Predictive Picture may be coded and decoded using intra prediction or inter prediction using at most two motion vectors and reference indices to predict the sample values of each block. Similarly, multiple-predictive pictures can use more than two reference pictures and associated metadata for the reconstruction of a single block.

[0044] Source pictures commonly may be subdivided spatially into a plurality of sample blocks (for example, blocks of 4×4, 8×8, 4×8, or 16×16 samples each) and coded on a block-by-block basis. Blocks may be coded predictively with reference to other (already coded) blocks as determined by the coding assignment applied to the blocks' respective pictures. For example, blocks of I pictures may be coded non-predictively or they may be coded predictively with reference to already coded blocks of the same picture (spatial prediction or intra prediction). Pixel blocks of P pictures may be coded non-predictively, via spatial prediction or via temporal prediction with reference to one previously coded reference pictures. Blocks of B pictures may be coded non-predictively, via spatial prediction or via temporal prediction with reference to one or two previously coded reference pictures.

[0045] A video may be captured as a plurality of source pictures (video pictures) in a temporal sequence. Intra-picture prediction (often abbreviated to intra prediction) makes use of spatial correlation in a given picture, and inter-picture prediction makes uses of the (temporal or other) correlation between the pictures. In an example, a specific picture under encoding/decoding, which is referred to as a current picture, is partitioned into blocks. When a block in the current picture is similar to a reference block in a previously coded and still buffered reference picture in the video, the block in the current picture can be coded by a vector that is referred to as a motion vector. The motion vector points to the reference block in the reference picture, and can have a third dimension identifying the reference picture, in case multiple reference pictures are in use.

[0046] The encoder component **106** may perform coding operations according to a predetermined video coding technology or standard, such as any described herein. In its operation, the encoder component **106** may perform various compression operations, including predictive coding operations that exploit temporal and spatial redundancies in the

input video sequence. The coded video data, therefore, may conform to a syntax specified by the video coding technology or standard being used.

[0047] FIG. **2**B is a block diagram illustrating example elements of the decoder component **122** in accordance with some embodiments. The decoder component **122** in FIG. **2**B is coupled to the channel **218** and the display **124**. In some embodiments, the decoder component **122** includes a transmitter coupled to the loop filter **256** and configured to transmit data to the display **124** (e.g., via a wired or wireless connection).

[0048] In some embodiments, the decoder component **122** includes a receiver coupled to the channel **218** and configured to receive data from the channel **218** (e.g., via a wired or wireless connection). The receiver may be configured to receive one or more coded video sequences to be decoded by the decoder component **122**. In some embodiments, the decoding of each coded video sequence is independent from other coded video sequences. Each coded video sequence may be received from the channel **218**, which may be a hardware/software link to a storage device which stores the encoded video data. The receiver may receive the encoded video data with other data, for example, coded audio data and/or ancillary data streams, that may be forwarded to their respective using entities (not depicted). The receiver may separate the coded video sequence from the other data. In some embodiments, the receiver receives additional (redundant) data with the encoded video. The additional data may be included as part of the coded video sequence(s). The additional data may be used by the decoder component **122** to decode the data and/or to more accurately reconstruct the original video data. Additional data can be in the form of, e.g., temporal, spatial, or SNR enhancement layers, redundant slices, redundant pictures, forward error correction codes, and so on.

[0049] In accordance with some embodiments, the decoder component **122** includes a buffer memory **252**, a parser **254** (also sometimes referred to as an entropy decoder), a scaler/inverse transform unit **258**, an intra picture prediction unit **262**, a motion compensation prediction unit **260**, an aggregator **268**, the loop filter unit **256**, a reference picture memory **266**, and a current picture memory **264**. In some embodiments, the decoder component **122** is implemented as an integrated circuit, a series of integrated circuits, and/or other electronic circuitry. The decoder component **122** may be implemented at least in part in software.

[0050] The buffer memory **252** is coupled in between the channel **218** and the parser **254** (e.g., to combat network jitter). In some embodiments, the buffer memory **252** is separate from the decoder component **122**. In some embodiments, a separate buffer memory is provided between the output of the channel **218** and the decoder component **122**. In some embodiments, a separate buffer memory is provided outside of the decoder component **122** (e.g., to combat network jitter) in addition to the buffer memory **252** inside the decoder component **122** (e.g., which is configured to handle playout timing). When receiving data from a store/forward device of sufficient bandwidth and controllability, or from an isosynchronous network, the buffer memory **252** may not be needed, or can be small. For use on best effort packet networks such as the Internet, the buffer memory **252** may be required, can be comparatively large and/or of adaptive size, and may at least partially be implemented in an operating system or similar elements outside of the decoder component **122**.

[0051] The parser **254** is configured to reconstruct symbols **270** from the coded video sequence. The symbols may include, for example, information used to manage operation of the decoder component **122**, and/or information to control a rendering device such as the display **124**. The control information for the rendering device(s) may be in the form of, for example, Supplementary Enhancement Information (SEI) messages or Video Usability Information (VUI) parameter set fragments (not depicted). The parser **254** parses (entropy-decodes) the coded video sequence. The coding of the coded video sequence can be in accordance with a video coding technology or standard, and can follow principles well known to a person skilled in the art, including variable length coding, Huffman coding, arithmetic coding with or without context sensitivity, and so forth. The parser **254** may extract from the coded video sequence, a set of subgroup parameters for at least one of the subgroups of pixels in the video decoder, based upon at least one parameter corresponding to the group. Subgroups can include Groups of Pictures (GOPs), pictures, tiles, slices, macroblocks, Coding Units (CUs), blocks, Transform Units (TUs), Prediction Units (PUs) and so forth. The parser **254** may also extract, from the coded video sequence, information such as transform coefficients, quantizer parameter values, motion vectors, and so forth.

[0052] Reconstruction of the symbols **270** can involve multiple different units depending on the type of the coded video picture or parts thereof (such as: inter and intra picture, inter and intra block), and other factors. Which units are involved, and how they are involved, can be controlled by the subgroup control information that was parsed from the coded video sequence by the parser **254**. The flow of such subgroup control information between the parser **254** and the multiple units below is not depicted for clarity.

[0053] The decoder component **122** can be conceptually subdivided into a number of functional units, and in some implementations, these units interact closely with each other and can, at least partly, be integrated into each other. However, for clarity, the conceptual subdivision of the functional units is maintained herein.

[0054] The scaler/inverse transform unit **258** receives quantized transform coefficients as well as control information (such as which transform to use, block size, quantization factor, and/or quantization scaling matrices) as symbol(s) **270** from the parser **254**. The scaler/inverse transform unit **258** can output blocks including sample values that can be input into the aggregator **268**. In some cases, the output samples of the scaler/inverse transform unit **258** pertain to an intra coded block; that is: a block that is not using predictive information from previously reconstructed pictures, but can use predictive information from previously reconstructed parts of the current picture. Such predictive information can be provided by the intra picture prediction unit **262**. The intra picture prediction unit **262** may generate a block of the same size and shape as the block under reconstruction, using surrounding already-reconstructed information fetched from the current (partly reconstructed) picture from the current picture memory **264**. The aggregator **268** may add, on a per sample basis, the prediction information the intra picture prediction unit **262** has gener-

ated to the output sample information as provided by the scaler/inverse transform unit **258**.

[0055] In other cases, the output samples of the scaler/inverse transform unit **258** pertain to an inter coded, and potentially motion-compensated, block. In such cases, the motion compensation prediction unit **260** can access the reference picture memory **266** to fetch samples used for prediction. After motion compensating the fetched samples in accordance with the symbols **270** pertaining to the block, these samples can be added by the aggregator **268** to the output of the scaler/inverse transform unit **258** (in this case called the residual samples or residual signal) so to generate output sample information. The addresses within the reference picture memory **266**, from which the motion compensation prediction unit **260** fetches prediction samples, may be controlled by motion vectors. The motion vectors may be available to the motion compensation prediction unit **260** in the form of symbols **270** that can have, for example, X, Y, and reference picture components. Motion compensation may also include interpolation of sample values as fetched from the reference picture memory **266**, e.g., when sub-sample exact motion vectors are in use, motion vector prediction mechanisms.

[0056] The output samples of the aggregator **268** can be subject to various loop filtering techniques in the loop filter unit **256**. Video compression technologies can include in-loop filter technologies that are controlled by parameters included in the coded video bitstream and made available to the loop filter unit **256** as symbols **270** from the parser **254**, but can also be responsive to meta-information obtained during the decoding of previous (in decoding order) parts of the coded picture or coded video sequence, as well as responsive to previously reconstructed and loop-filtered sample values. The output of the loop filter unit **256** can be a sample stream that can be output to a render device such as the display **124**, as well as stored in the reference picture memory **266** for use in future inter-picture prediction.

[0057] Certain coded pictures, once reconstructed, can be used as reference pictures for future prediction. Once a coded picture is reconstructed and the coded picture has been identified as a reference picture (by, for example, parser **254**), the current reference picture can become part of the reference picture memory **266**, and a fresh current picture memory can be reallocated before commencing the reconstruction of the following coded picture.

[0058] The decoder component **122** may perform decoding operations according to a predetermined video compression technology that may be documented in a standard, such as any of the standards described herein. The coded video sequence may conform to a syntax specified by the video compression technology or standard being used, in the sense that it adheres to the syntax of the video compression technology or standard, as specified in the video compression technology document or standard and specifically in the profiles document therein. Also, for compliance with some video compression technologies or standards, the complexity of the coded video sequence may be within bounds as defined by the level of the video compression technology or standard. In some cases, levels restrict the maximum picture size, maximum frame rate, maximum reconstruction sample rate (measured in, for example megasamples per second), maximum reference picture size, and so on. Limits set by levels can, in some cases, be further restricted through Hypothetical Reference Decoder (HRD) specifications and metadata for HRD buffer management signaled in the coded video sequence.

[0059] FIG. **3** is a block diagram illustrating the server system **112** in accordance with some embodiments. The server system **112** includes control circuitry **302**, one or more network interfaces **304**, a memory **314**, a user interface **306**, and one or more communication buses **312** for interconnecting these components. In some embodiments, the control circuitry **302** includes one or more processors (e.g., a CPU, GPU, and/or DPU). In some embodiments, the control circuitry includes field-programmable gate array(s), hardware accelerators, and/or integrated circuit(s) (e.g., an application-specific integrated circuit).

[0060] The network interface(s) **304** may be configured to interface with one or more communication networks (e.g., wireless, wireline, and/or optical networks). The communication networks can be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of communication networks include local area networks such as Ethernet, wireless LANs, cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Such communication can be unidirectional, receive only (e.g., broadcast TV), unidirectional send-only (e.g., CANbus to certain CANbus devices), or bi-directional (e.g., to other computer systems using local or wide area digital networks). Such communication can include communication to one or more cloud computing networks.

[0061] The user interface **306** includes one or more output devices **308** and/or one or more input devices **310**. The input device(s) **310** may include one or more of: a keyboard, a mouse, a trackpad, a touch screen, a data-glove, a joystick, a microphone, a scanner, a camera, or the like. The output device(s) **308** may include one or more of: an audio output device (e.g., a speaker), a visual output device (e.g., a display or monitor), or the like.

[0062] The memory **314** may include high-speed random-access memory (such as DRAM, SRAM, DDR RAM, and/or other random access solid-state memory devices) and/or non-volatile memory (such as one or more magnetic disk storage devices, optical disk storage devices, flash memory devices, and/or other non-volatile solid-state storage devices). The memory **314** optionally includes one or more storage devices remotely located from the control circuitry **302**. The memory **314**, or, alternatively, the non-volatile solid-state memory device(s) within the memory **314**, includes a non-transitory computer-readable storage medium. In some embodiments, the memory **314**, or the non-transitory computer-readable storage medium of the memory **314**, stores the following programs, modules, instructions, and data structures, or a subset or superset thereof:

- **[0063]** an operating system **316** that includes procedures for handling various basic system services and for performing hardware-dependent tasks;
- **[0064]** a network communication module **318** that is used for connecting the server system **112** to other computing devices via the one or more network interfaces **304** (e.g., via wired and/or wireless connections);
- **[0065]** a coding module **320** for performing various functions with respect to encoding and/or decoding

data, such as video data. In some embodiments, the coding module **320** is an instance of the coder component **114**. The coding module **320** including, but not limited to, one or more of:

[0066] a decoding module **322** for performing various functions with respect to decoding encoded data, such as those described previously with respect to the decoder component **122**; and

[0067] an encoding module **340** for performing various functions with respect to encoding data, such as those described previously with respect to the encoder component **106**; and

[0068] a picture memory **352** for storing pictures and picture data, e.g., for use with the coding module **320**. In some embodiments, the picture memory **352** includes one or more of: the reference picture memory **208**, the buffer memory **252**, the current picture memory **264**, and the reference picture memory **266**.

[0069] In some embodiments, the decoding module **322** includes a parsing module **324** (e.g., configured to perform the various functions described previously with respect to the parser **254**), a transform module **326** (e.g., configured to perform the various functions described previously with respect to the scalar/inverse transform unit **258**), a prediction module **328** (e.g., configured to perform the various functions described previously with respect to the motion compensation prediction unit **260** and/or the intra picture prediction unit **262**), and a filter module **330** (e.g., configured to perform the various functions described previously with respect to the loop filter **256**).

[0070] In some embodiments, the encoding module **340** includes a code module **342** (e.g., configured to perform the various functions described previously with respect to the source coder **202** and/or the coding engine **212**) and a prediction module **344** (e.g., configured to perform the various functions described previously with respect to the predictor **206**). In some embodiments, the decoding module **322** and/or the encoding module **340** include a subset of the modules shown in FIG. **3**. For example, a shared prediction module is used by both the decoding module **322** and the encoding module **340**.

[0071] Each of the above identified modules stored in the memory **314** corresponds to a set of instructions for performing a function described herein. The above identified modules (e.g., sets of instructions) need not be implemented as separate software programs, procedures, or modules, and thus various subsets of these modules may be combined or otherwise re-arranged in various embodiments. For example, the coding module **320** optionally does not include separate decoding and encoding modules, but rather uses a same set of modules for performing both sets of functions. In some embodiments, the memory **314** stores a subset of the modules and data structures identified above. In some embodiments, the memory **314** stores additional modules and data structures not described above.

[0072] Although FIG. **3** illustrates the server system **112** in accordance with some embodiments, FIG. **3** is intended more as a functional description of the various features that may be present in one or more server systems rather than a structural schematic of the embodiments described herein. In practice, items shown separately could be combined and some items could be separated. For example, some items shown separately in FIG. **3** could be implemented on single servers and single items could be implemented by one or more servers. The actual number of servers used to implement the server system **112**, and how features are allocated among them, will vary from one implementation to another and, optionally, depends in part on the amount of data traffic that the server system handles during peak usage periods as well as during average usage periods.

Example Coding Techniques

[0073] The coding processes and techniques described below may be performed at the devices and systems described above (e.g., the source device **102**, the server system **112**, and/or the electronic device **120**). In accordance with some embodiments, methods for inverse pre-filtering are described below.

[0074] Pre-filtering before video compression refers to applying a filter (e.g., a pre-filter) to a video stream before the video compression process begins. Pre-filtering can make a signal easier to compress by removing unnecessary or redundant information prior to the compression. However, the pre-filter may remove some of the useful information from the video stream, which in turn leads to distortion/degradation during reconstruction. As disclosed, an inverse pre-filter is an additional filter that is applied after pre-filtering, to recover information that was lost due to the pre-filtering.

[0075] FIGS. **4**A-**4**C illustrate an example of video encoding and decoding in accordance with some embodiments. FIG. **4**A illustrates the computation of a prediction block in accordance with some embodiments. In the example of FIG. **4**A, an intra prediction is performed on a current block **402** to generate a predicted block **404**. In some embodiments, an inter prediction is performed to generate the predicted block. The current block **402** includes a set of samples (e.g., pixel blocks) and the prediction block **404** includes a set of predictions that correspond to the set of samples. FIG. **4**B illustrates the computation of a residue block in accordance with some embodiments. As shown in FIG. **4**B, the prediction block **404** is subtracted from the current block **402** to generate a residue block **406** that includes a set of residues. For example, respective differences are calculated between each sample and the corresponding prediction. FIG. **4**C illustrates the computation of a reconstructed block in accordance with some embodiments. As shown in FIG. **4**C, the residue block **406** undergoes one or more transformations and quantization to generate a set of residual coefficients. The set of residual coefficients may be transmitted from an encoder component to a decoder component. The set of residual coefficients undergo a reverse quantization and reverse transformation to generate a reconstructed residue block **408**. The reconstructed residue block **408** is combined with the predicted block **404** (e.g., reconstructed residues of the reconstructed residue block **408** are added to predictions of the prediction block **404**) to generate a reconstructed block **410** corresponding to the current block **402**.

[0076] FIG. **5** illustrates example in-loop filtering stages according to some embodiments. In FIG. **5**, the in-loop filtering stages applied to a decoded frame **522** include a deblocking filter **524**, a constrained direction enhancement filter (CDEF) **526**, and a loop restoration filter **530**. In some embodiments, the filtered output frame is used as a reference frame for later frames (e.g., stored in a reference frame buffer **528**). The loop filtering methods may include any filtering process applied on the reconstructed samples (e.g., after adding residual to the prediction), including wiener

loop filtering, cross-component filtering, and CDEF. In some instances, the reconstructed sample filtered by loop filtering can be used as a reference sample for performing prediction within a picture. In some instances, the reconstructed sample filtered by loop filtering cannot be used as a reference sample.

[0077] A cross-component filtering method may use a co-located reconstructed sample and neighboring reconstructed samples from a first color component as input, to perform filtering of the current reconstruction sample of a second color component. A cross-component offset filtering method may use the co-located reconstructed sample and its neighboring reconstructed samples from a first color component as input, to derive an offset value that is added on the current sample of a second color component to adjust its reconstruction value. The first color component may refer to a luma color component, and the second color component may refer to a chroma color component. The first color component and second color component may be the same color component (e.g., a luma component).

[0078] The deblocking filter **524** may be applied across the transform block boundaries to remove block artifacts caused by the quantization error. In some embodiments, a filter length is determined based on the minimum transform block sizes on both sides. In some embodiments, finite impulse response (FIR) filters (e.g., low-pass filters) are used by the deblocking filter **524**. Edge detection may be used to disable the deblocking filter at transitions that contain a high variance signal (e.g., to avoid blurring an actual edge in the original image). In this way, a deblocking filtering method may be applied on reconstructions samples located close to block boundaries. The block boundaries may include a transform block boundary, a motion compensation block boundary, a coding block boundary, and/or a fixed block size boundary.

[0079] The CDEF **526** applies a non-linear de-ringing filter along particular (e.g., oblique) directions. The CDEF **526** may operate on an output of the deblocking filter **524**. The CDEF **526** may operate in 8×8 units. In some embodiments, 8 preset directions are defined by rotating and reflecting templates of preset directions. The decoder may use the reconstructed pixels to select the prevalent direction index. A primary filter may be applied along the selected direction, and a secondary filter may be applied along an offset direction (e.g., oriented 45° off the primary direction). In some embodiments, up to 8 groups of filter parameters are signaled (e.g., in a frame header). The groups of filter parameters may include the primary and secondary filter strength indexes of luma and chroma components. The CDEF **526** may apply filtering on reconstruction samples by identifying the direction of each block and then adaptively filtering with a high degree of control over the filter strength along the direction and across it.

[0080] In some embodiments, the loop restoration filter **530** is applied to reconstructed pixels after any prior in-loop filtering stages (e.g., the deblocking filter **524** and/or the CDEF **526**). The loop restoration filter **530** may be applied to loop restoration units (LRU), e.g., 64×64, 128×128, and/or 256×256 pixel blocks. Bypass filtering, a wiener filter (e.g., a wiener loop filtering method), and/or a self-guided filter may be applied to each LRU independently. A wiener loop filtering method may use a linear weighted sum of the current reconstruction sample and multiple spatially neighboring reconstruction samples as input to derive a modified value for the current reconstruction sample as the output.

[0081] FIGS. **6**A-**6**D illustrate example configurations for implementing an inverse pre-filter **618** in accordance with some embodiments. FIG. **6**A shows an example configuration **610** in accordance with some embodiments. In this example, the inverse pre-filter **618** is a postfilter that is enabled if the pre-filter **612** is enabled. The inverse pre-filter **618** is optimized based on original video data (org **611**), prior to a pre-filtering step via pre-filter **612**. The output of the pre-filter **612** comprises filtered samples (e.g., filt **613**), which are input into an encoder **614**. The encoder **614** outputs initial encoded pictures (rec **615**), which are input into one or more conventional in-loop filters **616**. The inverse pre-filter **618** is applied after the one or more conventional in-loop filters **616**. In the configuration **610**, the reconstructed samples (rec' **617**) after the conventional in-loop filter(s) **618** are stored in a decoded picture buffer (DPB) **620** and are used as the input for the inverse pre-filter **618**. The output from the inverse pre-filter **618** comprises filtered encoded pictures (rec" **619**).

[0082] FIG. **6**B illustrates another example configuration **630** in accordance with some embodiments. Here, the inverse pre-filter **618** is an additional in-loop filter that is enabled if the pre-filter **612** is enabled. In the configuration **630**, the inverse pre-filter **618** is optimized based on the original video data (org **631**) before the pre-filtering step via pre-filter **612**.

[0083] The output of the pre-filter **612** comprises filtered samples (e.g., filt **633**), which are input into the encoder **614**. The encoder **614** outputs initial encoded pictures (rec **635**), which are input into one or more conventional in-loop filters **616**. The inverse pre-filter **618** is applied after the one or more conventional in-loop filters **616**. The input of the inverse pre-filter **618** are the reconstructed samples after the conventional in-loop filters **616** (represented as rec' **637** in FIG. **6**B). The inverse pre-filter **618** outputs reconstructed samples (rec" **639**), which are stored in the DPB **620**.

[0084] FIG. **6**C illustrates another example configuration **650** in accordance with some embodiments. In FIG. **6**C, the inverse pre-filter **618** is a postfilter that is enabled if the pre-filter **612** is enabled. The inverse pre-filter **618** is optimized based on the original video data (org **651**) before a pre-filtering step via pre-filter **612**. The output of the pre-filter **612** comprises filtered samples (e.g., filt **653**), which are input into the encoder **614**. The encoder **614** outputs initial encoded pictures (rec **655**). The inverse pre-filter **618** is applied after the encoder **614**, in parallel with one or more conventional in-loop filters **616**. Thus, the reconstructed samples after the conventional in-loop filtering step (rec' **657**) are stored in the DPB **620**. The input to the inverse pre-filter **618** are the initial encoded pictures (rec **655**). The output from the inverse pre-filter **618** are filtered encoded pictures (rec" **659**)

[0085] FIG. **6**D illustrates another example configuration **670** in accordance with some embodiments. In FIG. **6**D, the inverse pre-filter **612** is a postfilter that is enabled if the pre-filter is enabled. In some embodiments, the inverse pre-filter **612** is optimized based on the original video data (org **671**) before a pre-filtering step via pre-filter **612**. The output of the pre-filter **612** comprises filtered samples (e.g., filt **673**), which are input into the encoder **614**. The encoder **614** outputs initial encoded pictures (rec **675**). The inverse pre-filter **618** is applied after one or more conventional

encoder in-loop filters **672**, in parallel with some other conventional in-loop filter(s) such as in-loop filter **674**. The reconstructed samples (rec' **677**) after the conventional in-loop filter **674** are stored in the DPB **620**. The input for the inverse pre-filter comprises first reconstructed samples (first rec **680**) after the initial set of one or more conventional encoder in-loop filters **672**.

[0086] FIG. **7**A is a flow diagram illustrating a method **700** of decoding video in accordance with some embodiments. The method **700** may be performed at a computing system (e.g., the server system **112**, the source device **102**, or the electronic device **120**) having control circuitry and memory storing instructions for execution by the control circuitry. In some embodiments, the method **700** is performed by executing instructions stored in the memory (e.g., the memory **314**) of the computing system.

[0087] The system receives (**702**) a video bitstream (e.g., a coded video sequence) comprising a current picture. The system obtains (**704**) a filtered current picture by applying at least one in-loop filter (e.g., in addition to a pre-filter applied prior to encoding the current picture) to the current picture. The system obtains (**706**) a reconstructed current picture by applying an inverse pre-filter to the filtered current picture. In this way, depending on the presence of a pre-filter, an inverse pre-filter is used to recover the information that was lost due to the pre-filtering step.

[0088] In some embodiments, the inverse pre-filter (e.g., inverse pre-filter **618**) is a postfilter that is enabled if the pre-filter is enabled. It is optimized based on the original input signal (before pre-filter), and it is applied after the conventional in-loop filter(s) if the latter exist(s). In other words, the reconstructed samples (e.g., rec' **617**) after the conventional in-loop filter are stored in the DPB, which is used as the input for the inverse pre-filter as shown in FIG. **6**A.

[0089] In some embodiments the inverse pre-filter is an offline trained filter. In some embodiments, the inverse pre-filter is online trained Wiener filter, and the coefficients of this filter are signalled in the bitstream. In some embodiments, the inverse pre-filter is Adaptive Loop Filter (ALF) that shares the same parameters (such as classification information etc.) with the original ALF, if it exists. In this case, only filter coefficients are different and require additional signalling. In some embodiments, the inverse pre-filter is an ALF that has all different parameters (such as classifications etc.) from the original ALF, if it exists. In this case all parameters such as classification information, filter coefficients, etc. are different and require additional signalling.

[0090] In some embodiments, the inverse pre-filter (e.g., inverse pre-filter **618**) is an additional in-loop filter that is enabled if the is pre-filter enabled. It is optimized based on the original input signal (before pre-filter), and it is applied after the conventional in-loop filter(s) if the latter exist(s). In other words, the reconstructed samples after the inverse pre-filter (e.g., rec" **639**) are stored in the DPB, as shown in FIG. **6**B. The input of the inverse pre-filter is the reconstructed samples (e.g., rec' **637**) after the conventional in-loop filter (e.g., in-loop filter **616**), as shown in FIG. **6**B.

[0091] In some embodiments, the inverse pre-filter is an offline trained filter. In some embodiments, the inverse pre-filter is online trained Wiener filter, and the coefficients of this filter are signalled in the bitstream. In some embodiments, the inverse pre-filter is Adaptive Loop Filter (ALF) that shares the same parameters (such as classification information etc.) with the original ALF, if it exists. In this case, only filter coefficients are different and require additional signalling. In some embodiments, the inverse pre-filter is an ALF that has all different parameters (such as classifications etc.) from the original ALF, if it exists. In this case all parameters such as classification information, filter coefficients, etc. are different and require additional signalling.

[0092] In some embodiments, the inverse pre-filter (e.g., inverse pre-filter **618**) is a postfilter that is enabled if the pre-filter is enabled. It is optimized based on the original input signal (before pre-filter) and it is applied after the encoder, in parallel with conventional in-loop filter(s) if the latter exist(s). In other words, the rec samples (rec' **657**) after the conventional in-loop filter are stored in DPB. The input for the proposed inverse pre-filter, in contrast to FIG. **6**A, is reconstructed samples (e.g., rec **655**) before conventional in-loop filters, as shown in FIG. **6**C.

[0093] In some embodiments, the inverse pre-filter is an offline trained filter. In some embodiments, the inverse pre-filter is online trained Wiener filter, and the coefficients of this filter are signalled in the bitstream. In some embodiments, the inverse pre-filter is Adaptive Loop Filter (ALF) that shares the same parameters (such as classification information etc.) with the original ALF, if it exists. In this case, only filter coefficients are different and require additional signalling. In some embodiments, the inverse pre-filter is an ALF that has all different parameters (such as classifications etc.) from the original ALF, if it exists. In this case all parameters such as classification information, filter coefficients, etc. are different and require additional signalling.

[0094] In some embodiments, the inverse pre-filter (e.g., inverse pre-filter **618**) is a postfilter that is enabled if the pre-filter is enabled. It is optimized based on the original input signal (before pre-filter) and it is applied after some subset of the conventional encoder in-loop filters (e.g., in-loop filter **672**) and in parallel to some other conventional in-loop filter(s) (e.g., in-loop filter **674**) if any of those two subsets exist. In other words, the rec samples (e.g., rec' **677**) after the conventional in-loop filter are stored in the DPB The input for the inverse pre-filter is reconstructed samples after a sub-set of conventional in loop filters (e.g., first rec **680**), as shown in FIG. **6**D.

[0095] In some embodiments, the inverse pre-filter is an offline trained filter. In some embodiments, the inverse pre-filter is online trained Wiener filter, and the coefficients of this filter are signalled in the bitstream. In some embodiments, the inverse pre-filter is Adaptive Loop Filter (ALF) that shares the same parameters (such as classification information etc.) with the original ALF, if it exists. In this case, only filter coefficients are different and require additional signalling. In some embodiments, the inverse pre-filter is an ALF that has all different parameters (such as classifications etc.) from the original ALF, if it exists. In this case all parameters such as classification information, filter coefficients, etc. are different and require additional signalling.

[0096] In some embodiments, an adaptive up-scale/down-scale filter or filter set can be combined with the pre-filter **618** in FIGS. **6**A to **6**D when the adaptive resolution changing feature is enabled during the encoding and decoding.

[0097] FIG. **7**B is a flow diagram illustrating a method **750** of encoding video in accordance with some embodiments. The method **750** may be performed at a computing system (e.g., the server system **112**, the source device **102**, or the electronic device **120**) having control circuitry and memory storing instructions for execution by the control circuitry. In some embodiments, the method **750** is performed by executing instructions stored in the memory (e.g., the memory **314**) of the computing system. In some embodiments, the method **750** is performed by a same system as the method **700** described above.

[0098] The system receives (**752**) video data (e.g., a source video sequence) comprising a set of pictures including a current picture. The system obtains (**754**) a filtered current picture by applying a pre-filter and at least one in-loop filter to the current picture. The system obtains (**756**) an encoded current picture by applying an inverse pre-filter to the filtered current picture As described previously, the encoding process may mirror the decoding processes described herein (e.g., applying inverse pre-filtering for image and video compression). For brevity, those details are not repeated here.

[0099] Although FIGS. **7**A and **7**B illustrate a number of logical stages in a particular order, stages which are not order dependent may be reordered and other stages may be combined or broken out. Some reordering or other groupings not specifically mentioned will be apparent to those of ordinary skill in the art, so the ordering and groupings presented herein are not exhaustive. Moreover, it should be recognized that the stages could be implemented in hardware, firmware, software, or any combination thereof.

[0100] Turning now to some example embodiments.

[0101] (A1) In one aspect, some embodiments include a method (e.g., the method **700**) of video decoding. In some embodiments, the method is performed at a computing system (e.g., the server system **112**) having memory and control circuitry. In some embodiments, the method is performed at a coding module (e.g., the coding module **320**). In some embodiments, the method is performed at a source coding component (e.g., the source coder **202**), a coding engine (e.g., the coding engine **212**), and/or an entropy coder (e.g., the entropy coder **214**). The method includes (i) receiving a video bitstream (e.g., a coded video sequence) comprising a current picture; (ii) obtaining a filtered current picture by applying at least one in-loop filter to the current picture; and (iii) obtaining a reconstructed current picture by applying an inverse pre-filter to the filtered current picture. For example, depending on the presence of a pre-filter, an inverse pre-filter is applied that targets to recover the information that was lost due to the pre-filtering step. In some embodiments, the inverse pre-filter is selectively applied in accordance with a determination that a pre-filter was applied to an input picture corresponding to the current picture. In some embodiments, when a pre-filter has been applied (e.g., as indicated in the video bitstream), an inverse pre-filter is applied to a reconstructed picture. In some embodiments, a syntax element is parsed from the video bitstream, the syntax element indicating whether a pre-filter operation was performed. The syntax element may be indicated in a high-level syntax (e.g., at a sequence level or picture level).

[0102] (A2) In some embodiments of A1, an output of the in-loop filter, without any inverse pre-filtering, is used for in-loop filtering. For example, the output of the in-loop filter (the filtered current picture) is stored in a DPB for used with in-loop filtering. As an example, the inverse pre-filter may be a postfilter that is enabled if the pre-filter is enabled and is optimized based on the original input signal (before the pre-filter is applied). The inverse pre-filter may be applied after the conventional in-loop filter(s) if the latter exists, e.g., as illustrated in FIG. **6**A.

[0103] (A3) In some embodiments of A1, the reconstructed current picture output by the inverse pre-filter is used for in-loop filtering. For example, the inverse pre-filter may be an additional in-loop filter that is enabled if the pre-filter is enabled, and it is optimized based on the original input signal (before pre-filtering). The inverse pre-filter may be applied after the conventional in-loop filter(s) if the latter exists, e.g., as illustrated in FIG. **6**B. Thus, the input of the proposed inverse pre-filter may be the reconstructed samples after the conventional in-loop filter.

[0104] (A4) In some embodiments of A1, the at least one in-loop filter comprises a plurality of in-loop filters, and wherein the inverse prefilter is arranged after a first in-loop filter of the plurality of in-loop filters and before a second in-loop filter of the plurality of in-loop filters. For example, the inverse pre-filter may be a postfilter that is enabled if a pre-filter is enabled, it is optimized based on the original input signal (before pre-filtering), and it is applied after some subset of the conventional encoder in-loop filters and in parallel to some other conventional in-loop filter(s) if any of those two subsets exist, e.g., as illustrated in FIG. **6**D. As an example, the first in-loop filter may be a time invariant filter (e.g., a deblocking filter) and the second in-loop filter may be a temporal filter.

[0105] (A5) In some embodiments of any of A1-A4, the inverse pre-filter is a fixed filter having fixed parameters. For example, the inverse pre-filter may be an offline-trained filter.

[0106] (A6) In some embodiments of any of A1-A5, one or more coefficients for the inverse pre-filter are signaled in the video bitstream. For example, the inverse pre-filter is an online-trained filter and the coefficients of this filter are signalled in the bitstream.

[0107] (A7) In some embodiments of A6, the inverse pre-filter is a Wiener filter. For example, the inverse pre-filter is an online-trained Wiener filter (or other type of time-invariant filter).

[0108] (A8) In some embodiments of A6, the inverse pre-filter is an adaptive loop filter (ALF).

[0109] (A9) In some embodiments of A8, the inverse pre-filter uses a same set of parameters as another ALF that is applied to one or more blocks of the current picture. For example, the inverse pre-filter is an ALF that shares the same parameters (such as classification information, etc.) with the original ALF, if such an ALF exists. In this case, only filter coefficients are different and require additional signalling.

[0110] (A10) In some embodiments of A8, the inverse pre-filter uses a first set of parameters that is different than a second set of parameters used by another ALF that is applied to one or more blocks of the current picture. For example, the inverse pre-filter is an ALF that has all different parameters (such as classification information, etc.) from the original ALF, if such an ALF exists. In this case, all parameters such as classification information, filter coefficients, etc. are different and require additional signalling.

[0111] (A11) In some embodiments of A10, the method further comprises parsing, from the video bitstream, the first

set of parameters. For example, the first set of parameters may be signaled at a sequence level, a picture level, or other level in the video bitstream.

[0112] (A12) In some embodiments of any of A1-A11, the inverse pre-filter comprises a rescaling filter. For example, an adaptive up-scale and/or down-scale filter (or filter set) can be combined with the inverse pre-filter when an adaptive resolution changing feature is enabled during the encoding and decoding. In some embodiments, the rescaling filter is conditionally enabled according to an adaptive resolution changing mode. In some embodiments, a syntax in the video bitstream indicates whether to enable the rescaling filter aspect of the inverse pre-filter. In some embodiments, the rescaling filter is applied prior to any inverse pre-filtering. In some embodiments, the rescaling filter is applied after any inverse pre-filtering.

[0113] (B1) In another aspect, some embodiments include a method (e.g., the method **750**) of video encoding. In some embodiments, the method is performed at a computing system (e.g., the server system **112**) having memory and control circuitry. In some embodiments, the method is performed at a coding module (e.g., the coding module **320**). The method includes: (i) receiving video data (e.g., a source video sequence) comprising a set of pictures including a current picture; (ii) obtaining a filtered current picture by applying a pre-filter and at least one in-loop filter to the current picture; and (iii) obtaining an encoded current picture by applying an inverse pre-filter to the filtered current picture.

[0114] (B2) In some embodiments of B1, the inverse pre-filter is a fixed filter having fixed parameters.

[0115] (B3) In some embodiments of B1 or B2, one or more coefficients for the inverse pre-filter are signaled in the video bitstream.

[0116] (B4) In some embodiments of B3, the inverse pre-filter is an adaptive loop filter (ALF).

[0117] (B5) In some embodiments of B4, the inverse pre-filter uses a same set of parameters as another ALF that is applied to one or more blocks of the current picture.

[0118] (B6) In some embodiments of any of B1-B5, the method further comprises signaling, via a video bitstream, whether to apply the inverse pre-filter at a decoding component.

[0119] (B7) In some embodiments of any of B1-B6, the method further comprises signaling, via a video bitstream, the encoded current picture.

[0120] (C1) In another aspect, some embodiments include a method of visual media data processing. In some embodiments, the method is performed at a computing system (e.g., the server system **112**) having memory and control circuitry. In some embodiments, the method is performed at a coding module (e.g., the coding module **320**). The method includes: (i) obtaining a source video sequence that comprises a plurality of frames and (ii) performing a conversion between the source video sequence and a video bitstream of visual media data according to a format rule. The video bitstream comprises a current picture. The format rule specifies that (a) a filtered current picture is to be obtained by applying at least one in-loop filter to the current picture and (b) a reconstructed current picture is to be obtained by applying an inverse pre-filter to the filtered current picture.

[0121] (D1) In another aspect, some embodiments include a method of video decoding. The method includes (i) receiving a video bitstream comprising a current picture; (ii) obtaining a filtered current picture by applying at least one in-loop filter to the current picture; (iii) storing the filtered current picture in a decoded picture buffer for use in in-loop filtering; and (iv) obtaining a reconstructed current picture by applying an inverse pre-filter to the current picture. For example, the inverse pre-filter is a postfilter that is enabled if a pre-filter is enabled, it is optimized based on the original input signal (before pre-filter), and it is applied after the encoder in parallel to conventional in-loop filter(s) if the latter exists, e.g., as shown in FIG. **6**C. As an example, the input for the inverse pre-filter is reconstructed samples before conventional in-loop filters are applied.

[0122] (D2) In some embodiments of D1, the inverse pre-filter is a fixed filter having fixed parameters.

[0123] (D3) In some embodiments of D1 or D2, one or more coefficients for the inverse pre-filter are signaled in the video bitstream. In some embodiments, the inverse pre-filter is a Wiener filter. In some embodiments, the inverse pre-filter is an ALF. In some embodiments, the inverse pre-filter uses a same set of parameters as another ALF that is applied to one or more blocks of the current picture. In some embodiments, the inverse pre-filter uses a first set of parameters that is different than a second set of parameters used by another ALF that is applied to one or more blocks of the current picture.

[0124] In another aspect, some embodiments include a computing system (e.g., the server system **112**) including control circuitry (e.g., the control circuitry **302**) and memory (e.g., the memory **314**) coupled to the control circuitry, the memory storing one or more sets of instructions configured to be executed by the control circuitry, the one or more sets of instructions including instructions for performing any of the methods described herein (e.g., A1-A12, B1-B7, C1, and D1-D3 above).

[0125] In yet another aspect, some embodiments include a non-transitory computer-readable storage medium storing one or more sets of instructions for execution by control circuitry of a computing system, the one or more sets of instructions including instructions for performing any of the methods described herein (e.g., A1-A12, B1-B7, C1, and D1-D3 above).

[0126] Unless otherwise specified, any of the syntax elements (e.g., indicators) described herein may be high-level syntax (HLS). As used herein, HLS is signaled at a level that is higher than a block level. For example, HLS may correspond to a sequence level, a frame level, a slice level, or a tile level. As another example, HLS elements may be signaled in a video parameter set (VPS), a sequence parameter set (SPS), a picture parameter set (PPS), an adaptation parameter set (APS), a slice header, a picture header, a tile header, and/or a CTU header.

[0127] It will be understood that, although the terms "first," "second," etc. may be used herein to describe various elements, these elements should not be limited by these terms. These terms are only used to distinguish one element from another. The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the claims. As used in the description of the embodiments and the appended claims, the singular forms "a," "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will also be understood that the term "and/or" as used herein refers to and encompasses any and all possible combinations of one or more of the associated

listed items. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

[0128] As used herein, the term "if" can be construed to mean "when" or "upon" or "in response to determining" or "in accordance with a determination" or "in response to detecting" that a stated condition precedent is true, depending on the context. Similarly, the phrase "if it is determined [that a stated condition precedent is true]" or "if [a stated condition precedent is true]" or "when [a stated condition precedent is true]" can be construed to mean "upon determining" or "in response to determining" or "in accordance with a determination" or "upon detecting" or "in response to detecting" that the stated condition precedent is true, depending on the context.

[0129] The foregoing description, for purposes of explanation, has been described with reference to specific embodiments. However, the illustrative discussions above are not intended to be exhaustive or limit the claims to the precise forms disclosed. Many modifications and variations are possible in view of the above teachings. The embodiments were chosen and described in order to best explain principles of operation and practical applications, to thereby enable others skilled in the art.

What is claimed is:

1. A method of video decoding performed at a computing system having memory and one or more processors, the method comprising:

receiving a video bitstream comprising a current picture;

obtaining a filtered current picture by applying at least one in-loop filter to the current picture; and

obtaining a reconstructed current picture by applying an inverse pre-filter to the filtered current picture.

2. The method of claim **1**, wherein an output of the in-loop filter, without any inverse pre-filtering, is used for in-loop filtering.

3. The method of claim **1**, wherein the reconstructed current picture output by the inverse pre-filter is used for in-loop filtering.

4. The method of claim **1**, wherein the at least one in-loop filter comprises a plurality of in-loop filters, and wherein the inverse pre-filter is arranged after a first in-loop filter of the plurality of in-loop filters and before a second in-loop filter of the plurality of in-loop filters.

5. The method of claim **1**, wherein the inverse pre-filter is a fixed filter having fixed parameters.

6. The method of claim **1**, wherein one or more coefficients for the inverse pre-filter are signaled in the video bitstream.

7. The method of claim **6**, wherein the inverse pre-filter is a Wiener filter.

8. The method of claim **6**, wherein the inverse pre-filter is an adaptive loop filter (ALF).

9. The method of claim **8**, wherein the inverse pre-filter uses a same set of parameters as another ALF that is applied to one or more blocks of the current picture.

10. The method of claim **8**, wherein the inverse pre-filter uses a first set of parameters that is different than a second set of parameters used by another ALF that is applied to one or more blocks of the current picture.

11. The method of claim **10**, further comprising parsing, from the video bitstream, the first set of parameters.

12. The method of claim **1**, wherein the inverse pre-filter comprises a rescaling filter.

13. A method of video encoding performed at a computing system having memory and one or more processors, the method comprising:

receiving video data comprising a set of pictures including a current picture;

obtaining a filtered current picture by applying at least one in-loop filter to the current picture; and

obtaining an encoded current picture by applying an inverse pre-filter to the filtered current picture.

14. The method of claim **13**, wherein the inverse pre-filter is a fixed filter having fixed parameters.

15. The method of claim **13**, wherein one or more coefficients for the inverse pre-filter are signaled in a video bitstream.

16. The method of claim **15**, wherein the inverse pre-filter is an adaptive loop filter (ALF).

17. The method of claim **16**, wherein the inverse pre-filter uses a same set of parameters as another ALF that is applied to one or more blocks of the current picture.

18. The method of claim **13**, further comprising signaling, via a video bitstream, whether to apply the inverse pre-filter at a decoding component.

19. The method of claim **13**, further comprising signaling, via a video bitstream, the encoded current picture.

20. A non-transitory computer-readable storage medium storing a video bitstream that is generated by a video encoding method, the video encoding method comprising:

receiving video data comprising a set of pictures including a current picture;

obtaining a filtered current picture by applying at least one in-loop filter to the current picture; and

obtaining an encoded current picture by applying an inverse pre-filter to the filtered current picture.

* * * * *