VIDEO CODING WITH ENHANCED SUPPORT FOR STREAM ADAPTATION AND SPLICING

[0001] This application claims the benefit of:

U.S. Provisional Patent Application No. 61/636,566, filed 20 April 2012;

U.S. Provisional Patent Application No. 61/643,100 filed 4 May 2012; and

U.S. Provisional Patent Application No. 61/667,371 filed 2 July 2012,

the entire content each of which is hereby incorporated by reference.

TECHNICAL FIELD

[0002] This disclosure relates to video coding and, more particularly, to techniques related to randomly accessing coded video data.

BACKGROUND

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

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

an inter-coded (P or B) slice of a picture may use spatial prediction with respect to reference samples in neighboring blocks in the same picture or temporal prediction with respect to reference samples in other pictures. Pictures may be referred to as frames, and reference pictures may be referred to a reference frames. A video sequence may also be referred to as a bitstream.

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

SUMMARY

[0006] This disclosure describes various techniques for enhanced support of stream adaptation and splicing based on clean random access (CRA) pictures. This disclosure proposes, instead of using a flag in the slice header to indicate that a broken link picture is present, that a distinct network abstraction layer (NAL) unit type can be used to indicate the presence of a broken link picture. In some implementations of the techniques of this disclosure, a first distinct NAL unit type may be used to indicate the presence of a broken link picture with leading pictures, while a second distinct NAL unit type indicates the presence of a broken link picture without leading pictures. Additionally, in some implementation, a third distinct NAL unit type may be used to indicate the presence of a broken link picture with decodable leading pictures.

[0007] In one example, a method for processing video data includes receiving a first network abstraction layer (NAL) unit comprising a portion of the video data and based on a NAL unit type of the first NAL unit, detecting a broken link picture.

[0008] In another example, a method for processing video data includes receiving a network abstraction layer (NAL) unit comprising a portion of the video data, determining that the NAL unit comprises a broken link picture and setting a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

[0009] In another example, a device for processing video data includes one or more processors configured to receive a first network abstraction layer (NAL) unit comprising a portion of the video data; and based on a NAL unit type of the first NAL unit, detect a broken link picture.

[0010] In another example, a device for processing video data includes one or more processors configured to receive a network abstraction layer (NAL) unit comprising a portion of the video data; determine that the NAL unit comprises a broken link picture; and set a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

[0011] In another example, a device for processing video data includes means for receiving a first network abstraction layer (NAL) unit comprising a portion of the video data; and means for detecting a broken link picture based on a NAL unit type of the first NAL unit.

[0012] In another example, a device for processing video data includes means for receiving a network abstraction layer (NAL) unit comprising a portion of the video data; means for determining that the NAL unit comprises a broken link picture; and means for setting a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

[0013] In another example, a computer-readable storage medium storing instructions that when executed by one or more processors cause the one or more processors to receive a first network abstraction layer (NAL) unit comprising a portion of the video data; and based on a NAL unit type of the first NAL unit, detect a broken link picture.

[0014] In another example, a computer-readable storage medium storing instructions that when executed by one or more processors cause the one or more processors to receive a network abstraction layer (NAL) unit comprising a portion of the video data; determine that the NAL unit comprises a broken link picture; and set a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

[0015] The details of one or more examples are set forth in the accompanying drawings and the description below. Other features, objects, and advantages will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

[0016] FIG. 1 is a block diagram illustrating an example video encoding and decoding system that may utilize the techniques described in this disclosure.

[0017] FIGS. 2 A and 2B are conceptual diagrams illustrating the clean random access picture concept.

[0018] FIG. 3 is a block diagram illustrating an example video encoder that may implement the techniques described in this disclosure.

[0019] FIG. 4 is a block diagram illustrating an example video decoder that may implement the techniques described in this disclosure.

[0020] FIG. 5 is a block diagram illustrating an example set of devices that form part of a network for communication of video data.

[0021] FIG. 6 is a diagram illustrating an example video sequence including a RAP picture in accordance with the techniques described in this disclosure.

[0022] FIG. 7 is a flowchart illustrating an example method for coding RAP pictures in accordance with one or more examples described in this disclosure.

[0023] FIG. 8 is a flowchart illustrating an example method for coding RAP pictures in accordance with one or more examples described in this disclosure.

[0024] FIG. 9 is a flowchart illustrating an example method for coding RAP pictures in accordance with one or more examples described in this disclosure.

[0025] FIG. 10 is a flowchart illustrating an example method for decoding a slice in accordance with one or more examples described in this disclosure.

[0026] FIG. 11 is a flowchart illustrating an example method for encoding a slice in accordance with one or more examples described in this disclosure.

[0027] FIG. 12 is a flowchart illustrating an example video processing technique of this disclosure.

DETAILED DESCRIPTION

[0028] This disclosure describes various techniques for enhanced support of stream adaptation and splicing based on clean random access (CRA) pictures. Random access generally refers to the decoding of a video bitstream starting from a coded picture that is

not the first coded picture in the bitstream. Random access to a bitstream may be desirable for many video applications, such as broadcasting and streaming. For example, random access can enable a user to switch between different channels, jump to specific parts of a video, or switch to a different bitstream for stream adaptation (e.g., adaptation of the bit rate, the frame rate, the spatial resolution, etc.). Random access can be enabled by inserting random access pictures, also referred to as random access point (RAP) pictures, in regular or irregular (but typically regular) intervals into the video bitstream.

[0029] It is contemplated that the techniques of this disclosure may be implemented in conjunction with a variety of different coding standards. For purposes of explanation, the techniques of this disclosure will be described primarily with reference to the emerging High Efficiency Video Coding (HEVC) standard. One intermediate draft of the HEVC standard, referred to as "HEVC Working Draft 6" or "WD6," is described in document JCTVC-H1003, Bross et al., "High efficiency video coding (HEVC) text specification draft 6," Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 8th Meeting: San Jose, California, USA, February 2012, which, as of 11 March 2013, is downloadable from http://phenix.int-evry.fr/jct/doc_end_user/documents/8_San%20Jose/wg11/JCTVC-H1003-v22.zip. The HEVC standard continues to evolve, and a newer draft of the standard referred to as "HEVC Working Draft 10," or "WD10," which is described in document JCTVC-L1003_v18, Bross et al., "High Efficiency Video Coding (HEVC) Text Specification Draft 10," Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 12th Meeting: Geneva, Switzerland, 14-23 January 2013, which, as of 11 March 2013, is downloadable from http://phenix.it-sudparis.eu/jct/doc_end_user/documents/12_Geneva/wg11/JCTVC-L1003-v18.zip. The entire content of each of WD6 and WD10 is hereby incorporated by reference.

[0030] An instantaneous decoding refresh (IDR) picture can be used for random access. However, because an IDR picture starts a coded video sequence and always cleans the decoded picture buffer (DPB), pictures following the IDR in the decoding order cannot use pictures decoded prior to the IDR picture as reference pictures. Consequently, bitstreams relying on IDR pictures for random access can have significantly lower coding efficiency. To improve the coding efficiency, the concept of CRA pictures was introduced to allow pictures that follow the CRA picture in decoding order but precede

the CRA picture in output order to use pictures decoded before the CRA picture as reference pictures (e.g. reference pictures for inter prediction).

[0031] Pictures that follow a CRA picture in decoding order but precede the CRA picture in output order are referred to as leading pictures of the CRA picture. The leading pictures of a CRA picture can be correctly decoded if the decoding starts from an IDR or CRA picture before the current CRA picture. However, the leading pictures of a CRA picture cannot be correctly decoded when random access from the CRA picture occurs. Accordingly, these leading pictures are typically discarded (i.e. removed from the bitsream) during the random access decoding. To prevent error propagation from reference pictures that may not be available depending on where the decoding starts, all pictures that follow a CRA picture both in decoding order and output order do not use any picture that precedes the CRA picture either in decoding order or output order (which includes the leading pictures) as reference pictures.

[0032] Random access functionalities similar to those described above are supported in H.264/AVC with the use of recovery points in supplemental enhancement information (SEI) messages. An H.264/AVC video decoder implementation may or may not support such functionality. In HEVC, a bitstream starting with a CRA picture is considered to be a conforming bitstream. When a bitstream starts with a CRA picture, the leading pictures of the CRA picture may refer to unavailable reference pictures and, therefore, may not be able to be correctly decoded. However, HEVC specifies that the leading pictures of the starting CRA picture are not output, hence the name "clean random access." For establishment of bitstream conformance, HEVC specifies a decoding process to generate unavailable reference pictures for decoding of the non-output leading pictures, but conforming decoder implementations do not have to follow that decoding process, as long as the decoder can generate identical output compared to when the decoding process is performed from the beginning of the coded video sequence.

[0033] In some versions of the emerging HEVC standard, a conforming bitstream may contain no IDR pictures at all, and consequently, may contain a subset of a coded video sequence or an incomplete coded video sequence. In HEVC, a coded video sequence is defined as a sequence of access units that consists, in decoding order, of an IDR access unit followed by zero or more non-IDR access units including all subsequent access units up to but not including any subsequent IDR access unit.

[0034] The concept of "CRA pictures with broken links" has been introduced in various HEVC proposals. Compared to the CRA concept, CRA pictures with broken links proposes to additionally allow CRA pictures that are not at the beginning of the bitstream to have non-decodable leading pictures like a CRA picture that begins the bitstream can have. In HEVC WD6, a CRA picture that starts a bitstream is allowed to be followed in decoding order (also referred to as bitstream order) by leading pictures that cannot be decoded due to missing prior reference pictures. However, a CRA picture that falls in the middle of a bitstream is not allowed to have such non-decodable leading pictures. It has been proposed that this constraint be removed by adding a "broken link" flag that indicates the potential presence of such non-decodable leading pictures.

[0035] The "broken link" flag has been proposed as picture-level information of CRA pictures in the slice header or some other place for picture-level information – such as the adaptation parameter set (APS). When the broken link flag is equal to 1, the bitstream is allowed to contain leading pictures of the CRA picture that are not decodable due to missing preceding reference pictures even though the bitstream starts with an IDR picture or a CRA picture earlier in bitstream order.

[0036] For a CRA picture with the "broken link" flag equal to 1, the CRA pictures are additionally allowed to have non-decodable leading pictures as mentioned above. Existing "CRA pictures with broken links" approaches, and other existing video random access approaches, are associated with various potential problems and shortcomings. For example, the signaling of the "broken link" flag in in the slice header or the APS requires an entity (e.g., a server, media-aware network element (MANE), or video editor/splicer) that changes a normal CRA picture to a broken-link CRA (BLC) picture when needed, to be capable of entropy encoding and to parse the slice header and/or the APS to encode the flag. This also requires an entity (e.g., a server, MANE, or video editor) that identifies a BLC picture when needed to be capable of entropy decoding and to parse the slice header and/or the APS to find the flag. BLC pictures are also sometimes referred to as broken link access (BLA) pictures, e.g., in more recent versions of the HEVC standard. In this disclosure, these terms can be considered to be interchangeable.

[0037] This disclosure describes techniques that may address the shortcomings described above. This disclosure proposes, instead of using a flag in the slice header to indicate that a BLC picture is present, that a distinct network abstraction layer (NAL)

unit type can be used to indicate the presence of a BLC picture. In some implementations of the techniques of this disclosure, a first distinct NAL unit type may be used to indicate the presence of a BLC (also known as BLA) picture with leading pictures, while a second distinct NAL unit type indicates the presence of a BLC picture without leading pictures. Additionally, in some implementation, a third distinct NAL unit type may be used to indicate the presence of a broken link picture with decodable leading pictures.

[0038] Coded video data is organized into NAL units, each of which is effectively a packet that contains an integer number of bytes. The first byte of each NAL unit is a header byte that contains an indication of the type of data in the NAL unit, and the remaining bytes contain payload data of the type indicated by the header. NAL units can generally be divided into two main categories, NAL units that contain video coding layer data and NAL units that do not include video coding layer data. These two categories of NAL units are referred to generally as VCL NAL units and non-VCL NAL units, respectively. Non-VLC NAL units include, for example, NAL unit types for supplemental enhancement information and NAL unit types for parameter set information.

[0039] According to the techniques of this disclosure, in one example, a NAL unit type equal to 2 (which is reserved in HEVC WD6) may be used to indicate the presence of a broken link picture, e.g., a BLC picture, otherwise referred to as a BLA picture. By using a dedicated NAL unit type, as described in this disclosure, an entity (e.g., a server, MANE, or video editor/splicer) can change a normal CRA picture, for which, if such picture does not start the bitstream, all the associated leading pictures must be decodable (i.e., can be correctly decoded), to be a BLC picture when needed without entropy encoding or decoding of slice header syntax, APS syntax, or other bitstream syntax. Additionally, by implementing the techniques of this disclosure, an entity (e.g., a server, MANE, or video editor) can identify a BLC picture when needed without entropy decoding of slice header syntax, APS syntax, or other bitstream syntax.

[0040] Using one or more distinct NAL unit types to indicate the presence of a broken link picture is one technique introduced in this disclosure. Additional techniques will also be introduced below. These additional techniques may be used alternatively to or in conjunction with the dedicated NAL units already introduced.

[0041] FIG. 1 is a block diagram illustrating an example video encoding and decoding system 10 that may utilize the techniques described in this disclosure. As illustrated in

FIG. 1, system 10 includes a source device 12 that generates encoded video data that a destination device 14 may decode later. Source device 12 and destination device 14 may comprise any of a wide range of devices, including desktop computers, notebook (i.e., laptop) computers, tablet computers, set-top boxes, telephone handsets such as so-called "smart" phones, so-called "smart" pads, televisions, cameras, display devices, digital media players, video gaming consoles, video streaming device, or the like. In some cases, source device 12 and destination device 14 may be equipped for wireless communication.

[0042] In some cases, video encoder 20 may encode a picture or pictures that are after a random access point or stream adaptation point, such as a temporal layer switching point. For example, this could be a switching point for adaptation of bit rate, frame rate or spatial resolution. One or more of these pictures may be leading pictures of a CRA picture in a video sequence. A decoder may decode the leading pictures of a CRA picture correctly if the decoding of the video sequence starts from a RAP picture before the current CRA picture in the video sequence. However, the leading pictures of a CRA picture cannot be correctly decoded when random access from the CRA picture occurs. For example, leading pictures may point to blocks for prediction reference that are unavailable. Accordingly, the leading picture may not be decodable at video decoder 30. Accordingly, a destination device 14 may typically discard these leading pictures during the random access decoding.

[0043] In another example, for a BLA, i.e., BLC, picture, encoder 20 may encode a flag, e.g., a no_output_of_prior_pics_flag (which may be more correctly referred to as a syntax element) so that none of the prior pictures in the DPB are output. In some examples, this flag, or syntax element, may be early in the slice header before entropy decoding so that it can be more easily decoded and the information is available earlier in the coding process. For example, so that less sophisticated devices, such as a MANE, may have access to the information without requiring an entropy decoder since, for example, the no_output_of_prior_pics_flag does not need to be entropy decoded. As an illustration, the no_output_of_prior_pics flag may be presented as a non-entropy coded syntax element such as, e.g., as a fixed length u(1) element, instead of an entropy coded syntax element, such as, e.g., a variable length ue(v) element. The no_output_of_prior_pics flag may be presented, for example, immediately after a first_slice_segment_in_pic flag, and before any entropy coded syntax elements.

[0044] In one example, video decoder 30 may mark pictures stored in a reference picture buffer as unused for reference when, for example, the use of these pictures may result in an incorrectly decoded picture. For example, a reference picture that precedes a BLA or BLC picture in decode order or output order may be unusable for reference for a leading picture that follows the BLA or BLC picture in decode order. Accordingly, the reference picture may be marked as unused for reference by video decoder 30 in response to receiving a reference picture that precedes a BLA or BLC picture in decode order or output order may be unusable for reference for a leading picture that follows the BLA or BLC picture in decode order.

[0045] In an example, a video encoder 20 may be configured to include NAL unit types assigned to indicate when BLA or BLC pictures have and do not have leading pictures. For example, in one standard, NAL unit types 16, BLA_W_LP (BLA with leading picture); 17, BLA_W_DLP (BLA with decodable leading picture); and 18, BLA_N_LP (BLA no leading picture) are included. A BLA picture having a NAL unit type equal to BLA_W_LP may have associated decodable or non-decodable leading pictures present in the bitstream. A BLA picture having a NAL unit type equal to BLA_W_DLP does not have associated non-decodable leading pictures present in the bitstream,but may have associated decodable leading pictures in the bitstream. A BLA picture having a NAL unit type equal to BLA_N_LP may not have associated leading pictures (either decodable or non-decodable) present in the bitstream.

[0046] In an example, video encoder 20 may process NAL unit types assigned to indicate when BLA pictures have and do not have leading pictures. For example, video encoder 20 may encode pictures according to one of a plurality of different network abstraction layer (NAL) unit types. The plurality of NAL unit types includes one or more of (1) a coded slice of a broken link access (BLA) picture, wherein the BLA picture is a BLA picture with associated leading pictures in a bitstream, (2) a coded slice of a BLA picture, wherein the BLA picture is a BLA picture with associated decodable leading pictures in the bitstream, and (3) a coded slice of a BLA picture, wherein the BLA picture is a BLA picture with no associated leading pictures in the bitstream.

[0047] Destination device 14 may receive the encoded video data. The destination device may decode the received data via a link 16. Link 16 may comprise any type of medium or device capable of moving the encoded video data from source device 12 to destination device 14. In one example, link 16 may comprise a communication medium to enable source device 12 to transmit encoded video data directly to destination device

14 in real-time. The encoded video data may be modulated according to a communication standard, such as a wireless communication protocol, and transmitted to destination device 14. The communication medium may comprise any wireless or wired communication medium, such as a radio frequency (RF) spectrum or one or more physical transmission lines. The communication medium may form part of a packet-based network, such as a local area network, a wide-area network, or a global network such as the Internet. The communication medium may include routers, switches, base stations, or any other equipment that may be useful to facilitate communication from source device 12 to destination device 14.

[0048] Alternatively, encoded data may be output from output interface 22 to a storage device 34. Similarly, encoded data may be accessed from storage device 34 by input interface. Storage device 34 may include any of a variety of distributed or locally accessed data storage media such as a hard drive, Blu-ray discs, DVDs, CD-ROMs, flash memory, volatile or non-volatile memory, or any other suitable digital storage media for storing encoded video data. In a further example, storage device 34 may correspond to a file server or another intermediate storage device that may hold the encoded video generated by source device 12. Destination device 14 may access stored video data from storage device 34 via streaming or download. The file server may be any type of server capable of storing encoded video data and transmitting that encoded video data to the destination device 14. Example file servers include a web server (e.g., for a website), an FTP server, network attached storage (NAS) devices, or a local disk drive. Destination device 14 may access the encoded video data through any standard data connection, including an Internet connection. This may include a wireless channel (e.g., a Wi-Fi connection), a wired connection (e.g., DSL, cable modem, etc.), or a combination of both that is suitable for accessing encoded video data stored on a file server. The transmission of encoded video data from storage device 34 may be a streaming transmission, a download transmission, or a combination of both.

[0049] The techniques of this disclosure are not necessarily limited to wireless applications or settings. The techniques may be applied to video coding in support of any of a variety of multimedia applications, such as over-the-air television broadcasts, cable television transmissions, satellite television transmissions, streaming video transmissions, e.g., via the Internet, encoding of digital video for storage on a data storage medium, decoding of digital video stored on a data storage medium, or other applications. In some examples, system 10 may be configured to support one-way or

two-way video transmission to support applications such as video streaming, video playback, video broadcasting, and/or video telephony.

[0050] In the example of FIG. 1, source device 12 includes a video source 18, video encoder 20 and an output interface 22. In some cases, output interface 22 may include a modulator/demodulator (modem) and/or a transmitter. In source device 12, video source 18 may include a source such as a video capture device, e.g., a video camera, a video archive containing previously captured video, a video feed interface to receive video from a video content provider, and/or a computer graphics system for generating computer graphics data as the source video, or a combination of such sources. As one example, if video source 18 is a video camera, source device 12 and destination device 14 may form so-called camera phones or video phones. However, the techniques described in this disclosure may be applicable to video coding in general, and may be applied to wireless and/or wired applications.

[0051] The captured, pre-captured, or computer-generated video may be encoded by video encoder 20. The encoded video data may be transmitted directly to destination device 14 via output interface 22 of source device 12. The encoded video data may also (or alternatively) be stored onto storage device 34 for later access by destination device 14 or other devices, for decoding and/or playback.

[0052] Destination device 14 includes an input interface 28, a video decoder 30, and a display device 32. In some cases, input interface 28 may include a receiver and/or a modem. Input interface 28 of destination device 14 receives the encoded video data over link 16. The encoded video data communicated over link 16, or provided on storage device 34, may include a variety of syntax elements generated by video encoder 20 for use by a video decoder, such as video decoder 30, in decoding the video data. Such syntax elements may be included with the encoded video data transmitted on a communication medium, stored on a storage medium, or stored on a file server.

[0053] In some examples, video decoder 30 of destination device 14 may decode a picture or pictures that are after a random access point or stream adaptation point, such as a temporal layer switching point. For example, this could be a switching point for adaptation of bit rate, frame rate (i.e., temporal layer switching point), or spatial resolution. One or more of these pictures may be leading pictures. The leading pictures cannot be correctly decoded when random access from a broken link (i.e., BLA or BLC) picture occurs.

[0054] In an example, to prevent error propagation from reference pictures that may not be available depending on where the decoding starts, video decoder 30 may not use any picture that precedes the broken link picture either in decoding order or output order (which includes the leading pictures) as reference pictures.

[0055] In various examples, video decoder 30 may mark all reference pictures in the DPB as unused for reference before decoding the BLA picture when decoding a BLA or BLC picture. For example, video decoder 30 may mark reference pictures in the decoded picture buffer (DPB) as unused for reference.

[0056] In another example, encoder 20 may include in the bitstream, and decoder 30 may receive, a flag or other syntax element, e.g., a no_output_of_prior_pics_flag, for a BLA picture to be decoded. The flag, when equal to 1, indicates that none of the prior pictures in the DPB are output for display. In particular, when no_output_of_prior_pics_flag is equal to 1, decoder 30 empties all picture storage buffers in the decoded picture buffer without output of the pictures they contain. In some examples, this flag or syntax element may be presented very early in the slice header before entropy decoding so that it can be more easily decoded, without the need for entropy coding, and the information is available earlier in the coding process. For example, so that less sophisticated devices, such as a MANE, may have access to the information without requiring a decoder since, for example, the no_output_of_prior_pics_flag does not need to be entropy decoded.

[0057] In another example, video decoder 30 may process NAL unit types assigned to indicate when BLA or BLC pictures have and do not have leading pictures. Again, as mentioned above, BLA pictures are conceptually the same as BLC pictures in that BLA and BLC pictures signify broken link CRA pictures, but different in terminology. In one example, video decoder 30 may decode pictures according to one of a plurality of different network abstraction layer (NAL) unit types. The plurality of NAL unit types includes one or more of (1) a coded slice of a broken link access (BLA) picture, wherein the BLA picture is a BLA picture with associated leading pictures in a bitstream, (2) a coded slice of a BLA picture, wherein the BLA picture is a BLA picture with associated decodable leading pictures in the bitstream, and (3) a coded slice of a BLA picture, wherein the BLA picture is a BLA picture with no associated leading pictures in the bitstream. In response to detecting a BLA picture, video decoder 30 may activate one or more new parameter sets. Parameter sets will be explained in more detail later in this disclosure.

[0058] In some examples, video decoder 30 is configured to mark all reference pictures as unused for reference if it detects a BLA NAL unit type. If a BLA NAL unit type is detected by decoder 30, the reference pictures are not valid for decoding the BLA picture or any pictures following the BLA picture in decoding or output order.

[0059] Display device 32 may be integrated with, or external to, destination device 14. In some examples, destination device 14 may include an integrated display device and also be configured to interface with an external display device. In other examples, destination device 14 may be a display device. In general, display device 32 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device.

[0060] Video encoder 20 and video decoder 30 may operate according to a video compression standard, such as the High Efficiency Video Coding (HEVC) standard presently under development, and may conform to the HEVC Test Model (HM). Alternatively, video encoder 20 and video decoder 30 may operate according to other proprietary or industry standards, such as the ITU-T H.264 standard, alternatively referred to as MPEG-4, Part 10, Advanced Video Coding (AVC), or extensions of such standards. The techniques of this disclosure, however, are not limited to any particular coding standard. Other examples of video compression standards include MPEG-2 and ITU-T H.263. The HM presumes several additional capabilities of video coding devices relative to existing devices according to, e.g., ITU-T H.264/AVC. For example, whereas H.264 provides nine intra-prediction encoding modes, the HM may provide as many as thirty-three intra-prediction encoding modes.

[0061] Although not shown in FIG. 1, in some aspects, video encoder 20 and video decoder 30 may each be integrated with an audio encoder and decoder, and may include appropriate MUX-DEMUX units, or other hardware and software, to handle encoding of both audio and video in a common data stream or separate data streams. If applicable, in some examples, MUX-DEMUX units may conform to the ITU H.223 multiplexer protocol, or other protocols such as the user datagram protocol (UDP).

[0062] Video encoder 20 and video decoder 30 each may be implemented as any of a variety of suitable encoder circuitry, such as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic, software, hardware, firmware or any combinations thereof. When the techniques are implemented partially in software, a device may store

instructions for the software in a suitable, non-transitory computer-readable medium and execute the instructions in hardware using one or more processors to perform the techniques of this disclosure. Each of video encoder 20 and video decoder 30 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined encoder/decoder (CODEC) in a respective device.

[0063] In general, the working model of the HM describes that a video frame or picture may be divided into a sequence of treeblocks or largest coding units (LCU) that include both luma and chroma samples. A treeblock has a similar purpose as a macroblock of the H.264 standard. A slice includes a number of consecutive treeblocks in coding order. A video frame or picture may be partitioned into one or more slices. Each treeblock may be split into coding units (CUs) according to a quadtree. For example, a treeblock, as a root node of the quadtree, may be split into four child nodes, and each child node may in turn be a parent node and be split into another four child nodes. A final, unsplit child node, as a leaf node of the quadtree, comprises a coding node, i.e., a coded video block. Syntax data associated with a coded bitstream may define a maximum number of times a treeblock may be split, and may also define a minimum size of the coding nodes.

[0064] A CU includes a coding node and prediction units (PUs) and transform units (TUs) associated with the coding node. A size of the CU corresponds to a size of the coding node and must be square in shape. The size of the CU may range from 8x8 pixels up to the size of the treeblock with a maximum of 64x64 pixels or greater. Each CU may contain one or more PUs and one or more TUs. Syntax data associated with a CU may describe, for example, partitioning of the CU into one or more PUs. Partitioning modes may differ between whether the CU is skip or direct mode encoded, intra-prediction mode encoded, or inter-prediction mode encoded. PUs may be partitioned to be non-square in shape. Syntax data associated with a CU may also describe, for example, partitioning of the CU into one or more TUs according to a quadtree. A TU can be square or non-square in shape.

[0065] The HEVC standard allows for transformations according to TUs, which may be different for different CUs. The TUs are typically sized based on the size of PUs within a given CU defined for a partitioned LCU, although this may not always be the case. The TUs are typically the same size or smaller than the PUs. In some examples, residual samples corresponding to a CU may be subdivided into smaller units using a quadtree structure known as "residual quad tree" (RQT). The leaf nodes of the RQT

may be referred to as transform units (TUs). Pixel difference values associated with the TUs may be transformed to produce transform coefficients, which may be quantized.

[0066] In general, a PU includes data related to the prediction process. For example, when the PU is intra-mode encoded, the PU may include data describing an intra-prediction mode for the PU. As another example, when the PU is inter-mode encoded, the PU may include data defining a motion vector for the PU. The data defining the motion vector for a PU may describe, for example, a horizontal component of the motion vector, a vertical component of the motion vector, a resolution for the motion vector (e.g., one-quarter pixel precision or one-eighth pixel precision), a reference picture to which the motion vector points, and/or a reference picture list (e.g., List 0, List 1, or List C) for the motion vector.

[0067] In general, a TU is used for the transform and quantization processes. A given CU having one or more PUs may also include one or more transform units (TUs). Following prediction, video encoder 20 may calculate residual values corresponding to the PU. The residual values comprise pixel difference values that may be transformed into transform coefficients, quantized, and scanned using the TUs to produce serialized transform coefficients for entropy coding. This disclosure typically uses the term "video block" to refer to a coding node of a CU. In some specific cases, this disclosure may also use the term "video block" to refer to a treeblock, i.e., LCU, or a CU, which includes a coding node and PUs and TUs.

[0068] A video sequence typically includes a series of video frames or pictures. A group of pictures (GOP) generally comprises a series of one or more of the video pictures. A GOP may include syntax data in a header of the GOP, a header of one or more of the pictures, or elsewhere, that describes a number of pictures included in the GOP. Each slice of a picture may include slice syntax data that describes an encoding mode for the respective slice. Video encoder 20 typically operates on video blocks within individual video slices in order to encode the video data. A video block may correspond to a coding node within a CU. The video blocks may have fixed or varying sizes, and may differ in size according to a specified coding standard.

[0069] As an example, the HM supports prediction in various PU sizes. Assuming that the size of a particular CU is 2Nx2N, the HM supports intra-prediction in PU sizes of 2Nx2N or NxN, and inter-prediction in symmetric PU sizes of 2Nx2N, 2NxN, Nx2N, or NxN. The HM also supports asymmetric partitioning for inter-prediction in PU sizes of 2NxnU, 2NxnD, nLx2N, and nRx2N. In asymmetric partitioning, one direction of a CU

is not partitioned, while the other direction is partitioned into 25% and 75%. The portion of the CU corresponding to the 25% partition is indicated by an "n" followed by an indication of "Up," "Down," "Left," or "Right." Thus, for example, "2NxnU" refers to a 2Nx2N CU that is partitioned horizontally with a 2Nx0.5N PU on top and a 2Nx1.5N PU on bottom.

[0070] In this disclosure, "NxN" and "N by N" may be used interchangeably to refer to the pixel dimensions of a video block in terms of vertical and horizontal dimensions, e.g., 16x16 pixels or 16 by 16 pixels. In general, a 16x16 block will have 16 pixels in a vertical direction (y = 16) and 16 pixels in a horizontal direction (x = 16). Likewise, an NxN block generally has N pixels in a vertical direction and N pixels in a horizontal direction, where N represents a nonnegative integer value. The pixels in a block may be arranged in rows and columns. Moreover, blocks need not necessarily have the same number of pixels in the horizontal direction as in the vertical direction. For example, blocks may comprise NxM pixels, where M is not necessarily equal to N.

[0071] Following intra-predictive or inter-predictive coding using the PUs of a CU, video encoder 20 may calculate residual data for the TUs of the CU. The PUs may comprise pixel data in the spatial domain (also referred to as the pixel domain) and the TUs may comprise coefficients in the transform domain following application of a transform, e.g., a discrete cosine transform (DCT), an integer transform, a wavelet transform, or a conceptually similar transform to residual video data. The residual data may correspond to pixel differences between pixels of the unencoded picture and prediction values corresponding to the PUs. Video encoder 20 may form the TUs including the residual data for the CU, and then transform the TUs to produce transform coefficients for the CU.

[0072] Following any transforms to produce transform coefficients, video encoder 20 may perform quantization of the transform coefficients. Quantization generally refers to a process in which transform coefficients are quantized to possibly reduce the amount of data used to represent the coefficients, providing further compression. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, an *n*-bit value may be rounded down to an *m*-bit value during quantization, where *n* is greater than *m*.

[0073] In some examples, video encoder 20 may utilize a predefined scan order to scan the quantized transform coefficients to produce a serialized vector that can be entropy encoded. In other examples, video encoder 20 may perform an adaptive scan. After

scanning the quantized transform coefficients to form a one-dimensional vector, video encoder 20 may entropy encode the one-dimensional vector, e.g., according to context adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), syntax-based context-adaptive binary arithmetic coding (SBAC), Probability Interval Partitioning Entropy (PIPE) coding or another entropy encoding methodology. Video encoder 20 may also entropy encode syntax elements associated with the encoded video data for use by video decoder 30 in decoding the video data.

[0074] To perform CABAC, video encoder 20 may assign a context within a context model to a symbol to be transmitted. The context may relate to, for example, whether neighboring values of the symbol are non-zero or not. To perform CAVLC, video encoder 20 may select a variable length code for a symbol to be transmitted. Codewords in VLC may be constructed such that relatively shorter codes correspond to more probable symbols, while longer codes correspond to less probable symbols. In this way, the use of VLC may achieve a bit savings over, for example, using equal-length codewords for each symbol to be transmitted. The probability determination may be based on a context assigned to the symbol.

[0075] Video encoder 20 can be configured to code video data in a hierarchical manner. For example, video encoder 20 may hierarchically categorize video data as including a plurality of layers, a sequence of pictures within a given layer, a picture within a sequence, slices within a picture, and blocks (e.g., macroblocks or coding tree units) within a slice. A video parameter set (VPS) may signal infrequently changing parameters for a plurality of sequences across respective layers. Sequence parameter sets (SPSs) may be used to signal infrequently changing parameters for a sequence of pictures, and picture parameter sets (PPSs) may be used to signal infrequently changing parameters for individual pictures.

[0076] As introduced above, this disclosure describes techniques for enhanced support of stream adaptation and splicing based on CRA pictures. A CRA picture is a picture coded using intra coding techniques, such that decoding of the CRA pictures is not dependent on information from other pictures.

[0077] Random access generally refers to the ability to begin decoding a video bitstream starting from a coded picture that is not the first coded picture in the bitstream. Random access to a bitstream of coded video data may be desirable in many video applications, such as broadcasting and streaming. Streaming of video data over a network may be performed, e.g., using Dynamic Adaptive Streaming over HTTP

(DASH). For example, viewers of video data may wish to switch between different channels, jump to specific parts of a video, perform a so-called "trick mode" such as fast forward or rewind, or switch to a different bitstream (e.g., a different representation in DASH) for stream adaptation (e.g., adaptation of the bit rate (such as to accommodate fluctuating network bandwidth), frame rate, spatial resolution, etc.).

[0078] A random access feature can be enabled by inserting random access pictures or random access points, many times in regular intervals, into a bitstream of video data. An IDR unit can be used for random access. However, because an IDR unit typically starts a coded video sequence and cleans a DPB, pictures following the IDR unit in decoding order cannot use pictures decoded prior to the IDR unit as references. Consequently, bitstreams relying on IDR units for random access sometimes have significantly lower coding efficiency. To improve the coding efficiency associated with IDR units, the concept of CRA pictures was introduced in HEVC to allow pictures that follow the CRA picture in decoding order but precede it in output order to use pictures decoded before the CRA picture as references.

[0079] FIGS. 2A and 2B are conceptual diagrams illustrating the concept of CRA pictures. FIG. 2A shows a sequence of pictures 202 where the order from left to right indicates the order in which the pictures are decoded. Each picture also has a picture order count (POC) value signifying a display order for the pictures. The POC value is shown relative to CRA picture 204, such that negative values indicate pictures that are displayed before CRA picture 204 and positive values indicate pictures that are displayed after CRA picture 204. Sequence of pictures 202 includes CRA picture 204. Sequence of pictures 202 further includes a series of lagging pictures 206, a series of leading pictures 208, and a series of normal pictures 210. Lagging pictures 206 are prior to CRA picture 204 in the decoding order but follow it in the display order (i.e. the POC of lagging pictures 206 is higher than the POC of CRA picture 204). Leading pictures 208 follow CRA picture 204 in the decoding order but precede CRA picture 204 in the output order. Normal pictures 210 follow CRA picture 204 in both the decoding order and the output order.

[0080] FIG. 2B shows a sequence of pictures 203. Sequence of pictures 203 comprises the same pictures as sequence of pictures 202, but sequence of pictures 203 shows the pictures in display order, such that the ordering of the pictures from left to right corresponds to the order in which the pictures will be output. Pictures that follow a CRA picture in decoding order but precede the CRA picture in output order are referred

to as leading pictures of the CRA picture (e.g. leading pictures 208). The leading pictures of a CRA picture can be correctly decoded if the decoding starts from an IDR unit or CRA picture before the current CRA picture. However, the leading pictures of a CRA picture cannot be correctly decoded when random access from the CRA picture occurs. Thus, these leading pictures are typically discarded during the random access decoding. To prevent error propagation from reference pictures that may not be available depending on where the decoding starts, all pictures that follow a CRA picture both in decoding order and output order shall not use any picture that precedes the CRA picture either in decoding order or output order (which includes the leading pictures) as a reference.

[0081] Similar random access functionalities are supported in H.264/AVC with the recovery point supplemental enhancement information (SEI) message. An H.264/AVC decoder implementation may or may not support the functionality. In HEVC, a bitstream starting with a CRA picture is considered as a conforming bitstream. When a bitstream starts with a CRA picture, the leading pictures of the CRA picture may refer to unavailable reference pictures and hence cannot be correctly decoded. However, HEVC specifies that the leading pictures of the starting CRA picture are not output, hence the name "clean random access." For establishment of bitstream conformance requirement, HEVC specifies a decoding process to generate unavailable reference pictures for decoding of the non-output leading pictures. However, conforming decoder implementations do not have to follow that decoding process, as long as the decoder can they generate identical output compared to when the decoding process is performed from the beginning of the coded video sequence.

[0082] It is worth noting that in HEVC a conforming bitstream may contain no IDR units at all, and consequently may contain a subset of a coded video sequence or an incomplete coded video sequence. In HEVC, a coded video sequence is defined as a sequence of access units that consist, in decoding order, of an IDR access unit followed by zero or more non-IDR access units including all subsequent access units up to but not including any subsequent IDR access units.

[0083] The concept of "CRA pictures with broken links" was introduced in JCTVC-I0404 (available from http://phenix.int-evry.fr/jct/doc_end_user/documents/9_Geneva/wg11/JCTVC-I0404-v1.zip). Compared to the CRA concept included in HEVC WD6, it is proposed in JCTVC-I0404 to additionally allow CRA pictures that are not at the beginning of the bitstream to have

non-decodable leading pictures like a CRA picture that begins the bitstream can have. In HEVC WD6, a CRA picture that starts a bitstream is allowed to be followed (in bitstream order, also referred to as decoding order) by leading pictures that cannot be decoded due to missing prior reference pictures. However, a CRA picture that falls in the middle of a bitstream, or somewhere other than the start of the bitstream, is not allowed to have such non-decodable leading pictures. It is proposed in JCTVC-I0404 to remove this constraint, by adding a "broken link" flag that would indicate the potential presence of such non-decodable leading pictures.

[0084] The "broken link" flag is proposed in JCTVC-I0404 as picture-level information of CRA pictures in the slice header or some other place for picture-level information – such as the adaptation parameter set (APS). When the flag is equal to 1, the bitstream would be allowed to contain leading pictures of the CRA picture that are not decodable due to missing preceding reference pictures even though the bitstream starts with an IDR picture or a CRA picture earlier in bitstream order.

[0085] For a CRA picture with the "broken link" flag equal to 1, in addition to being allowed to have non-decodable leading pictures as mentioned above, a picture order count (POC) most significant bit (MSB) may be set to 0, it contains a no_output_of_prior_pics_flag that acts in the same way as for an IDR picture, and a random_access_pic_id that acts in the same way as the idr_pic_id of IDR pictures. Also, the current idr_pic_id (as in HEVC WD 6) is renamed to random_access_pic_id and its constraints should be made to apply to both CRA pictures and IDR pictures rather than only to IDR pictures. Like an IDR picture, a CRA picture with broken_link_flag equal to 1 could activate a different sequence parameter set (SPS), change the picture size, etc.

[0086] The "CRA pictures with broken links" approach in JCTVC-I0404 and other existing video random access approaches might, in some instances, be associated with several problems or shortcomings. For example, the signaling of the "broken link" flag in in the slice header or the APS requires an entity (e.g., a server, MANE, or video editor/splicer) that changes a normal CRA picture to be a BLC picture, when needed, to be capable of entropy coding and to parse the slice header and/or the APS to encode the flag. This also requires the entity that identifies a BLC picture when needed to be capable of entropy decoding and to parse the slice header and/or the APS to find the flag.

[0087] As another example, a broken-link CRA (BLC) picture is allowed to not activate an SPS, picture parameter set (PPS), or APS (when referred to by the picture) if the parameter set ID is the same as the active SPS, PPS or APS. However, because a BLC picture is usually originated from a different bitstream than the previous picture in decoding order, the BLC picture generally uses different SPS raw bit sequence payloads (RBSPs), PPS RBSPs, and APS RBSPs, and it is likely that both the BLC picture and the previous picture in decoding order refer to (directly or indirectly) the same value of SPS or PPS ID. It is also possible for them to refer to the same value of APS ID. Using the active SPS, PPS, or APS for the previous picture in decoding may in some instances cause decoding of the BLC picture and the following pictures (not just the leading pictures) to be incorrect.

[0088] As another example, without including the random_access_pic_id and no_output_of_prior_pics_flag in normal CRA pictures requires an entity that changes a normal CRA picture to be a BLC picture, when needed, to be capable of entropy encoding and to parse the slice header and/or the APS to encode the syntax elements. As yet another example, since a BLC picture is usually originated from a different bitstream than the previous picture in decoding order, if no_output_of_prior_pics_flag is equal to 0, the decoded picture buffer may overflow; thus all the following pictures in decoding order may be decoded incorrectly or the decoder may even crash.

[0089] As another example, when a part of a first bitstream and a part of a second bitstream, wherein the part of the second bitstream starts from a CRA picture picA, are spliced or concatenated, it is possible not to change the CRA picture to a BLC picture but rather keep picA as a CRA picture in case, when decoding starts from the previous CRA picture or IDR picture or BLC picture, the decoding quality of the leading pictures of the CRA picture picA is acceptable though not perfect (e.g., when the checksums of one or more of the leading pictures do not match with the checksums signaled in the decoded picture hash SEI messages as in HVEC WD6). However, there lacks a mechanism to indicate the above information in the bitstream.

[0090] This disclosure describes techniques that may, in some instances, improve upon some of the shortcomings discussed above. The various techniques described herein may be implemented either individually or in combination with other described techniques. The following techniques are based on the concept of CRA pictures with broken links as set forth in JCTVC-I0404.

[0091] According to one technique, instead of using a flag in the slice header to indicate a CRA picture with a broken link (i.e. a BLC picture), a distinct network abstraction layer (NAL) unit type (e.g., NAL unit type equal to 2 which is reserved in HEVC WD6) may be used to indicate the NAL unit belongs to a broken link picture, which may be referred to as BLC or BLA picture. One BLC picture may comprises more than one NAL units. Such a technique may be less complex for a network entity because the technique may not require entropy encoding or decoding for the entity to change a normal CRA picture (e.g. by changing a NAL unit type) to be a BLC picture, when needed. Additionally, it may not require entropy coding for the entity to identify a BLC picture when needed. For a normal CRA picture, which does not start a bitstream, all the associated leading pictures may need to be decodable (i.e., can be correctly decoded).

[0092] According to another technique, instead of allowing a BLC picture to activate a different SPS, a BLC picture is required to activate an SPS, even if the SPS ID referred to by the BLC picture (indirectly through a PPS as in HEVC WD6 or any other means, e.g., indirectly through a group parameter set as described in JCTVC-I0338, or directly, e.g., when the SPS ID is directly included in the slice header, or indirectly through a buffering period SEI message) is the same as the SPS ID of the active SPS for the previous picture in decoding order. This is because a BLC picture was typically from a different bitstream than the previous picture in decoding order, and typically different SPS RBSPs are applied, with identical or different SPS IDs.

[0093] A BLC picture is also required to activate a PPS, even if the PPS ID referred to by the BLC picture (indirectly through a group parameter set as described in JCTVC-I0338, or directly, e.g., when the PPS ID is directly included in the slice header as in HEVC WD6) is the same as the PPS ID of the active PPS for the previous picture in decoding order. This is again because a BLC picture was typically from a different bitstream than the previous picture in decoding order, and typically different PPS RBSPs are applied, with identical or different PPS IDs.

[0094] A BLC picture is also required to activate an APS if it refers to an APS, even if the APS ID referred to by the BLC picture is the same as the APS ID of the active PPS for the previous picture in decoding order. This is again because a BLC picture was typically from a different bitstream than the previous picture in decoding order, and typically different APS RBSPs are applied, with identical or different APS IDs.

[0095] As used in this disclosure, a BLC picture can be defined as a coded picture for which the NAL unit type is the one for BLC picture (e.g., NAL unit type equal to 2 that is reserved in HEVC WD6). A BLC picture may be considered to be a special type of CRA picture. The same decoding process for a non-BLC CRA picture when it starts the bitstream and the associated leading pictures is applied for decoding a BLC picture and the associated leading pictures, even if the BLC picture is not the first picture in the bitstream. Alternatively, BLC pictures may be excluded from CRA pictures, i.e., a BLC picture may be considered not a CRA picture. In that case, the same decoding process for a CRA picture when it starts the bitstream and the associated leading pictures is applied for decoding a BLC picture and the associated leading pictures, even if the BLC picture is not the first picture in the bitstream. In the following description of this document, it is assumed that this alternative applies. A BLC access unit may be defined as an access unit in which the coded picture is a BLC picture. The definition of coded video sequence may be changed as follows to be: a sequence of access units that includes, in decoding order, an IDR access unit or a BLC access unit followed by zero or more non-IDR and non-BLC access units including all subsequent access units up to but not including any subsequent IDR or BLC access unit.

[0096] According to another technique, instead of having either or both of a random_access_pic_id and a no_output_of_prior_pics_flag for only IDR pictures and BLC pictures as in JCTVC-I0404, either of both of the two fields may always be present for all IDR pictures, BLC pictures, and all CRA pictures. For each CRA picture, no_output_of_prior_pics_flag may be required to be equal to 0. Such a technique may in some instances make it easier for a network entity to change a CRA picture to a BLC picture, when needed. Alternatively, for a BLC picture, it may be required that no_output_of_prior_pics_flag is equal to 1. Alternatively, each BLC picture may not have a no_output_of_prior_pics_flag signaled, but the picture output behavior may be the same as if it had no_output_of_prior_pics_flag equal to 1. Alternatively, each BLC picture may have no_output_of_prior_pics_flag signaled, but the picture output behavior may be the same as if it had no_output_of_prior_pics_flag equal to 1, regardless of the value of the signaled no_output_of_prior_pics_flag.

[0097] According to another technique, the POC MSB value for a BLC picture may be signaled, e.g., in the slice header. If it is signaled, the value would still be considered as equal to 0 in the decoding process, regardless of the value. Alternatively, the signaled POC MSB value is used in the decoding process, but then the splicer needs to check and

possibly change the value to be consistent with the POC values of earlier pictures in decoding order.

[0098] According to another technique, when a part of a first bitstream and a part of a second bitstream, wherein the part of the second bitstream starts from a CRA picture picA, are spliced or concatenated, the splicer can keep picA as a CRA picture in case, when decoding starts from the previous CRA picture or IDR picture or BLC picture, the decoding quality of the leading pictures of the CRA picture picA is acceptable though not perfect (e.g., when the checksums of one or more of the leading pictures do not match with the checksums signaled in the decoded picture hash SEI messages as in HVEC WD6). An indication of the above information may be signaled in the bitstream. The information may be signaled through an indication associated with picA, e.g., as a flag in the NAL unit header or the slice header or the referred APS, or an SEI message associated with picA. The flag may be named exact_match_flag, the value 1 indicates that the checksum of each leading picture associated with picA matches the checksum signaled in the decoded picture hash SEI message, if present, and the value 0 indicates that the checksum of each leading picture associated with picA may or may not match the checksum signaled in the decoded picture hash SEI message, if present.

[0099] FIG. 3 is a block diagram illustrating an example video encoder 20 that may implement the techniques described in this disclosure. Video encoder 20 may be configured to output video to post processing entity 27. Post processing entity 27 is intended to represent an example of a video entity, such as a MANE or splicing/editing device, that may process encoded video data from video encoder 20. In some instances, post processing entity may be an example of a network entity. In some video encoding systems, post-processing entity 27 and video encoder 20 may be parts of separate devices, while in other instances, the functionality described with respect to post-processing entity 27 may be performed by the same device that comprises video encoder 20.

[0100] Video encoder 20 may perform intra- and inter-coding of video blocks within video slices. Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame or picture. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames or pictures of a video sequence. Intra-mode (I mode) may refer to any of several spatial based compression modes. Inter-modes, such as uni-directional prediction (P

mode) or bi-prediction (B mode), may refer to any of several temporal-based compression modes.

[0101] In the example of FIG. 3, video encoder 20 includes a partitioning unit 35, prediction processing unit 41, filter unit 63, picture memory 64, summer 50, transform processing unit 52, quantization unit 54, and entropy encoding unit 56. Prediction processing unit 41 includes motion estimation unit 42, motion compensation unit 44, and intra prediction processing unit 46. For video block reconstruction, video encoder 20 also includes inverse quantization unit 58, inverse transform processing unit 60, and summer 62. Filter unit 63 is intended to represent one or more loop filters such as a deblocking filter, an adaptive loop filter (ALF), and a sample adaptive offset (SAO) filter. Although filter unit 63 is illustrated in FIG. 3 as being an in loop filter, in other configurations, filter unit 63 may be implemented as a post loop filter.

[0102] As illustrated in FIG. 3, video encoder 20 receives video data, and partitioning unit 35 partitions the data into video blocks. This partitioning may also include partitioning into slices, tiles, or other larger units, as wells as video block partitioning, e.g., according to a quadtree structure of LCUs and CUs. Video encoder 20 generally illustrates the components that encode video blocks within a video slice to be encoded. The slice may be divided into multiple video blocks (and possibly into sets of video blocks referred to as tiles). Prediction processing unit 41 may select one of a plurality of possible coding modes, such as one of a plurality of intra coding modes or one of a plurality of inter coding modes, for the current video block based on error results (e.g., coding rate and the level of distortion). Prediction processing unit 41 may provide the resulting intra- or inter-coded block to summer 50 to generate residual block data and to summer 62 to reconstruct the encoded block for use as a reference picture.

[0103] As discussed above, in some cases, video encoder 20 may encode a random access point or stream adaptation point, such as a temporal layer switching point, such as a BLA or BLC picture. For example, the encoding may occur within entropy encoding unit 56, which may perform both entropy and non-entropy encoding. One or more of these pictures may be leading pictures of the CRA picture. The leading pictures of a CRA picture may be correctly decoded if the decoding starts from a RAP picture before the current CRA picture. However, the leading pictures of a CRA picture cannot be correctly decoded when random access from the CRA picture occurs. For example, leading pictures may point to blocks for prediction reference that are unavailable. Accordingly, the leading picture may not be decodable at video decoder 30.

Accordingly, these leading pictures are typically discarded during the random access decoding.

[0104] In an example, video encoder 20 may provide a flag in the slice header, e.g., a no_output_of_prior_pics_flag or syntax element so that none of the prior pictures, i.e., pictures prior to the BLA or BLC picture, in the DPB are output. In some examples, this flag (or syntax element) may be early in the slice header before entropy encoding so that it can be more easily decoded at decoder 30, for example, and the information may be available earlier in the coding process. The syntax element or flag may be encoded, e.g., in a slice header for a BLA or BLC picture, by entropy encoding unit 56 (which may perform non-entropy encoding). This may be useful, for example, for intermediate devices such as a MANE, so that the information provided by the syntax element or flag may be available to the intermediate device without entropy decoding, although it may also be helpful to a decoder to have access to such information prior to entropy decoding.

[0105] For example, video encoder 20 (e.g. entropy encoding unit 56) may include a syntax element that indicates that a picture storage buffer is emptied without outputting any pictures from the picture storage buffer. The syntax element, when set, may cause pictures prior to a current picture in decoding order and residing in the picture storage buffer at the time of the decoding of the current picture to be emptied from the picture storage buffer without being output. In some examples, the syntax element may be one of a plurality of syntax elements. Additionally, the plurality of syntax elements may include entropy coded slice header syntax elements and non-entropy coded slice header syntax elements. In an example, the syntax element that indicates that a picture storage buffer is emptied without outputting any pictures from the picture storage buffer is included in the slice header before any entropy coded slice header syntax element so that the syntax element itself is not entropy coded.

[0106] In some examples, the syntax element may be a no_output_of_prior_pics_flag and the no_output_of_prior_pics_flag may be included in the slice header immediately after a first_slice_in_pic_flag. The first_slice_in_pic_flag may be a flag that indicates whether a slice is the first slice, in decoding order, of a picture.

[0107] In an example, a video encoder 20 may be configured to include NAL unit types assigned to indicate when BLA or BLC pictures have and do not have leading pictures. For example, in one standard, NAL unit types 16, BLA_W_LP (BLA with leading picture); 17, BLA_W_DLP (BLA with decodable leading picture); and 18, BLA_N_LP

(BLA no leading picture) are included. These NAL unit types may be encoded by entropy encoding unit 56 (which may perform non-entropy encoding). Accordingly, based on NAL unit type, a decoder may know when a BLA picture has a leading picture and when the leading picture is not decodable, e.g., when decoding starts from an earlier RAP picture. Accordingly, this information might be used to determine when leading pictures may be marked as unused for reference, which may trigger the decoder to mark reference pictures in the decoded picture buffer as unused for reference.

[0108] Intra prediction processing unit 46 within prediction processing unit 41 may perform intra-predictive coding of the current video block relative to one or more neighboring blocks in the same frame or slice as the current block to be coded to provide spatial compression. Motion estimation unit 42 and motion compensation unit 44 within prediction processing unit 41 perform inter-predictive coding of the current video block relative to one or more blocks for prediction reference in one or more reference pictures to provide temporal compression.

[0109] Motion estimation unit 42 may be configured to determine the inter-prediction mode for a video slice according to a predetermined pattern for a video sequence. The predetermined pattern may designate video slices in the sequence as P slices, B slices or GPB slices. Motion estimation unit 42 and motion compensation unit 44 may be highly integrated, but are illustrated separately for conceptual purposes. Motion estimation, performed by motion estimation unit 42, is the process of generating motion vectors, which estimate motion for video blocks. A motion vector, for example, may indicate the displacement of a PU of a video block within a current video frame or picture relative to a predictive block within a reference picture.

[0110] A predictive block is a block that is found to closely match the PU of the video block to be coded in terms of pixel difference, which may be determined by sum of absolute difference (SAD), sum of square difference (SSD), or other difference metrics. In some examples, video encoder 20 may calculate values for sub-integer pixel positions of reference pictures stored in picture memory 64. For example, video encoder 20 may interpolate values of one-quarter pixel positions, one-eighth pixel positions, or other fractional pixel positions of the reference picture. Therefore, motion estimation unit 42 may perform a motion search relative to the full pixel positions and fractional pixel positions and output a motion vector with fractional pixel precision.

[0111] Motion estimation unit 42 calculates a motion vector for a PU of a video block in an inter-coded slice by comparing the position of the PU to the position of a

predictive block of a reference picture. The reference picture may be selected from a first reference picture list (List 0) or a second reference picture list (List 1), each of which identify one or more reference pictures stored in picture memory 64. Motion estimation unit 42 sends the calculated motion vector to entropy encoding unit 56 and motion compensation unit 44.

[0112] Motion compensation, performed by motion compensation unit 44, may involve fetching or generating the predictive block based on the motion vector determined by motion estimation, possibly performing interpolations to sub-pixel precision. Upon receiving the motion vector for the PU of the current video block, motion compensation unit 44 may locate the predictive block to which the motion vector points in one of the reference picture lists. Video encoder 20 forms a residual video block by subtracting pixel values of the predictive block from the pixel values of the current video block being coded, forming pixel difference values. The pixel difference values form residual data for the block, and may include both luma and chroma difference components. Summer 50 represents the component or components that perform this subtraction operation. Motion compensation unit 44 may also generate syntax elements associated with the video blocks and the video slice for use by video decoder 30 in decoding the video blocks of the video slice.

[0113] Intra-prediction unit 46 may intra-predict a current block, as an alternative to the inter-prediction performed by motion estimation unit 42 and motion compensation unit 44, as described above. In particular, intra-prediction unit 46 may determine an intra-prediction mode to use to encode a current block. In some examples, intra-prediction unit 46 may encode a current block using various intra-prediction modes, e.g., during separate encoding passes, and intra-prediction unit 46 (or mode select unit 40, in some examples) may select an appropriate intra-prediction mode to use from the tested modes. For example, intra-prediction unit 46 may calculate rate-distortion values using a rate-distortion analysis for the various tested intra-prediction modes, and select the intra-prediction mode having the best rate-distortion characteristics among the tested modes. Rate-distortion analysis generally determines an amount of distortion (or error) between an encoded block and an original, unencoded block that was encoded to produce the encoded block, as well as a bit rate (that is, a number of bits) used to produce the encoded block. Intra-prediction unit 46 may calculate ratios from the distortions and rates for the various encoded blocks to determine which intra-prediction mode exhibits the best rate-distortion value for the block.

[0114] In any case, after selecting an intra-prediction mode for a block, intra-prediction unit 46 may provide information indicative of the selected intra-prediction mode for the block to entropy encoding unit 56. Entropy encoding unit 56 may encode the information indicating the selected intra-prediction mode in accordance with the techniques of this disclosure. Video encoder 20 may include in the transmitted bitstream configuration data. The bitstream may include a plurality of intra-prediction mode index tables and a plurality of modified intra-prediction mode index tables (also referred to as codeword mapping tables), definitions of encoding contexts for various blocks, and indications of a most probable intra-prediction mode, an intra-prediction mode index table, and a modified intra-prediction mode index table to use for each of the contexts.

[0115] After prediction processing unit 41 generates the predictive block for the current video block via either inter-prediction or intra-prediction, video encoder 20 forms a residual video block by subtracting the predictive block from the current video block. The residual video data in the residual block may be included in one or more TUs and applied to transform processing unit 52. Transform processing unit 52 transforms the residual video data into residual transform coefficients using a transform, such as a discrete cosine transform (DCT) or a conceptually similar transform. Transform processing unit 52 may convert the residual video data from a pixel domain to a transform domain, such as a frequency domain.

[0116] Transform processing unit 52 may send the resulting transform coefficients to quantization unit 54. Quantization unit 54 quantizes the transform coefficients to further reduce bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. The degree of quantization may be modified by adjusting a quantization parameter. In some examples, quantization unit 54 may then perform a scan of the matrix including the quantized transform coefficients. Alternatively, entropy encoding unit 56 may perform the scan.

[0117] Following quantization, entropy encoding unit 56 entropy encodes the quantized transform coefficients. For example, entropy encoding unit 56 may perform context adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), syntax-based context-adaptive binary arithmetic coding (SBAC), probability interval partitioning entropy (PIPE) coding or another entropy encoding methodology or technique. Following the entropy encoding by entropy encoding unit 56, the encoded bitstream may be transmitted to video decoder 30, or archived for later transmission or

retrieval by video decoder 30. Entropy encoding unit 56 may also entropy encode the motion vectors and the other syntax elements for the current video slice being coded.

[0118] Inverse quantization unit 58 and inverse transform processing unit 60 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual block in the pixel domain for later use as a reference block of a reference picture. Motion compensation unit 44 may calculate a reference block by adding the residual block to a predictive block of one of the reference pictures within one of the reference picture lists. Motion compensation unit 44 may also apply one or more interpolation filters to the reconstructed residual block to calculate sub-integer pixel values for use in motion estimation. Summer 62 adds the reconstructed residual block to the motion compensated prediction block produced by motion compensation unit 44 to produce a reference block for storage in picture memory 64. The reference block may be used by motion estimation unit 42 and motion compensation unit 44 as a reference block to inter-predict a block in a subsequent video frame or picture.

[0119] FIG. 4 is a block diagram illustrating an example video decoder 30 that may implement the techniques described for enhanced support of stream adaptation and splicing based on broken link RAP pictures described above. In the example of FIG. 4, video decoder 30 includes an entropy decoding unit 80, prediction processing unit 81, inverse quantization unit 86, inverse transformation unit 88, summer 90, filter unit 91, and picture memory 92. Prediction processing unit 81 includes motion compensation unit 82 and intra prediction processing unit 84. Video decoder 30 may, in some examples, perform a decoding pass generally reciprocal to the encoding pass described with respect to video encoder 20 from FIG. 3.

[0120] During the decoding process, video decoder 30 receives an encoded video bitstream that represents video blocks of an encoded video slice and associated syntax elements from video encoder 20. Video decoder 30 may receive the encoded video bitstream from a network entity 29. Network entity 29 may, for example, be a server, a MANE, a video editor/splicer, or other such device configured to implement one or more of the techniques described above. Network entity 29 may or may not include a video encoder 20. As described above, some of the techniques described in this disclosure may be implemented by network entity 29 prior to network 29 transmitting the encoded video bitstream to video decoder 30. In some video decoding systems, network entity 29 and video decoder 30 may be parts of separate devices, while in other

instances, the functionality described with respect to network entity 29 may be performed by the same device that comprises video decoder 30.

[0121] Network entity 29 represents an example of a video processing device configured to receive a first network abstraction layer (NAL) unit comprising a portion of video data and based on a NAL unit type of the first NAL unit, detect a broken link picture. Network entity 29 also can represent an example of a video processing device configured to receive a network abstraction layer (NAL) unit comprising a portion of the video data; determine that the NAL unit comprises a broken link picture; and set a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

[0122] As discussed above, in some examples, video decoder 30 may decode a broken link RAP picture, such as a BLA picture, or pictures that are after a random access point or stream adaptation point, such as a temporal layer switching point, such as leading pictures. The leading pictures cannot be correctly decoded when random access from the broken link RAP picture occurs.

[0123] In an example, to prevent error propagation from reference pictures that may not be available depending on where the decoding starts, video decoder 30 may not use any picture that precedes the broken link RAP picture either in decoding order or output order (which includes the leading pictures) as reference pictures. For example, prediction processing unit 81 may not use any picture stored in picture memory 92 that precedes the broken link RAP picture either in decoding order or output order (which includes the leading pictures) as reference pictures.

[0124] In various examples, video decoder 30 may mark all reference pictures in the DPB as unused for reference before decoding the BLA picture as unused for reference. For example, entropy decoding unit 80, which may perform entropy and non-entropy decoding, may mark reference pictures in picture memory 92, sometimes referred to as the decoded picture buffer (DPB), as unused for reference. Video decoder 30 (e.g. entropy decoding unit 80) may determine that a current picture is a BLA or BLC picture and mark reference pictures in a picture storage buffer as unused for reference prior to decoding the BLA or BLC picture. Determining that a current picture is a BLA or BLC picture may include determining that the current picture is a CRA picture and determining that the current picture is a RAP picture. The current CRA picture is a BLA picture when the current picture is both a CRA picture and a RAP picture. In

some examples, this determination by decoder 30 might be made based on the picture having a BLA NAL unit type.

[0125] In another example, when decoding a BLA picture, decoder 30 may receive a flag or syntax element in the encoded bitstream, e.g., a no_output_of_prior_pics_flag, so that none of the prior pictures in the DPB are output. In some examples, this flag may be presented early in a slice header, of a slice of a BLA picture, before entropy decoding so that it can be more easily decoded and the information is available earlier in the coding process. The flag or syntax element may be decoded by entropy decoding unit 80, which may perform both entropy and non-entropy decoding. Placing the flag or other syntax element early in the slice header before entropy encoding may allow for less sophisticated devices, such as a MANE to have access to the information without requiring an entropy decoder since, for example, the no_output_of_prior_pics_flag does not need to be entropy decoded in this example.

[0126] In an example, video decoder 30 may place a syntax element, e.g., a no_output_of_prior_pics_flag, in the bitstream for a decoder to receive. The syntax element may indicate that a picture storage buffer is emptied without outputting any pictures from the picture storage buffer. The syntax element, when set, may cause pictures prior to a current picture in decoding order and residing in the picture storage buffer at the time of the decoding of the current picture to be emptied from the picture storage buffer without being output. In some examples, the syntax element may be one of a plurality of syntax elements. Additionally, the plurality of syntax elements may include one or more entropy coded slice header syntax elements and one or more non-entropy coded slice header syntax elements. In an example, the syntax element that indicates that a picture storage buffer is emptied without outputting any pictures from the picture storage buffer is included in the slice header, e.g., as a u(1) element), before any entropy coded slice header syntax element, e.g., before any ue(v) element. In some examples, the syntax element may be a no_output_of_prior_pics_flag and the no_output_of_prior_pics_flag may be included in the slice header immediately after a first_slice_in_pic_flag and before any entropy coded elements. The first_slice_in_pic_flag may be a flag that indicates whether a slice is the first slice, in decoding order, of a picture.

[0127] In another example, video decoder 30 may process NAL unit types assigned to indicate when BLA pictures have and do not have leading pictures. Video decoder 30 may be configured to include NAL unit types assigned to indicate when BLA pictures

have and do not have leading pictures. For example, entropy decoding unit 80, which may perform entropy and non-entropy decoding, may process the NAL unit types.

[0128] In one example, video decoder 30 (e.g., entropy decoding unit 80) may decode pictures according to one of a plurality of different network abstraction layer (NAL) unit types. The plurality of NAL unit types includes one or more of (1) a coded slice of a broken link picture (e.g., referred to as BLA or BLC), wherein the broken link picture is a broken link picture with associated leading pictures (either decodable or non-decodable) in a bitstream, (2) a coded slice of a broken link picture, wherein the broken link picture is a BLA picture with associated decodable leading pictures in the bitstream, and (3) a coded slice of a broken link picture, wherein the broken link picture is a broken link picture with no associated leading pictures in the bitstream. In an example, the leading picture comprises a picture that precede a random access picture (RAP) in a display order but follows the random access picture in decoding order.

[0129] Entropy decoding unit 80 of video decoder 30 entropy decodes the bitstream to generate quantized coefficients, motion vectors, and other syntax elements. Entropy decoding unit 80 forwards the motion vectors and other syntax elements to prediction processing unit 81. Video decoder 30 may receive the syntax elements at the video slice level and/or the video block level.

[0130] When the video slice is coded as an intra-coded (I) slice, intra prediction processing unit 84 of prediction processing unit 81 may generate prediction data for a video block of the current video slice based on a signaled intra prediction mode and data from previously decoded blocks of the current frame or picture. When the video frame is coded as an inter-coded (i.e., B, P or GPB) slice, motion compensation unit 82 of prediction processing unit 81 produces blocks for prediction reference for a video block of the current video slice based on the motion vectors and other syntax elements received from entropy decoding unit 80. The blocks for prediction reference may be produced from one of the reference pictures within one of the reference picture lists. Video decoder 30 may construct the reference frame lists, List 0 and List 1, using default construction techniques based on reference pictures stored in picture memory 92.

[0131] Motion compensation unit 82 determines prediction information for a video block of the current video slice by parsing the motion vectors and other syntax elements, and uses the prediction information to produce the blocks for prediction reference for the current video block being decoded. For example, motion compensation unit 82 uses

some of the received syntax elements to determine a prediction mode (e.g., intra- or inter-prediction) used to code the video blocks of the video slice, an inter-prediction slice type (e.g., B slice, P slice, or GPB slice), construction information for one or more of the reference picture lists for the slice, motion vectors for each inter-encoded video block of the slice, inter-prediction status for each inter-coded video block of the slice, and other information to decode the video blocks in the current video slice. When pictures in the DPB are marked as unused for reference, there are no reference pictures available. Accordingly, video decoder 30 will not be able to decode a leading picture referencing an earlier reference picture for inter-prediction.

[0132] Motion compensation unit 82 may also perform interpolation based on interpolation filters. Motion compensation unit 82 may use interpolation filters as used by video encoder 20 during encoding of the video blocks to calculate interpolated values for sub-integer pixels of reference blocks. In this case, motion compensation unit 82 may determine the interpolation filters used by video encoder 20 from the received syntax elements and use the interpolation filters to produce blocks for prediction reference.

[0133] Inverse quantization unit 86 inverse quantizes, i.e., de-quantizes, the quantized transform coefficients provided in the bitstream and decoded by entropy decoding unit 80. The inverse quantization process may include use of a quantization parameter calculated by video encoder 20 for each video block in the video slice to determine a degree of quantization and, likewise, a degree of inverse quantization that should be applied. Inverse transform processing unit 88 applies an inverse transform, e.g., an inverse DCT, an inverse integer transform, or a conceptually similar inverse transform process, to the transform coefficients in order to produce residual blocks in the pixel domain.

[0134] After motion compensation unit 82 generates the predictive block for the current video block based on the motion vectors and other syntax elements, video decoder 30 forms a decoded video block by summing the residual blocks from inverse transform processing unit 88 with the corresponding blocks for prediction reference generated by motion compensation unit 82. Summer 90 represents the component or components that perform this summation operation. If desired, loop filters (either in the coding loop or after the coding loop) may also be used to smooth pixel transitions, or otherwise improve the video quality. Filter unit 91 is intended to represent one or more loop filters such as a deblocking filter, an adaptive loop filter (ALF), and a sample adaptive

offset (SAO) filter. Although filter unit 91 is illustrated in FIG. 4 as being an in loop filter, in other configurations, filter unit 91 may be implemented as a post loop filter. The decoded video blocks in a given frame or picture are then stored in picture memory 92, which stores reference pictures used for subsequent motion compensation. Picture memory 92 also stores decoded video for later presentation on a display device, such as display device 32 of FIG. 1.

[0135] In this manner, video decoder 30 of FIG. 4 represents an example of a video decoder configured to receive a first network abstraction layer (NAL) unit comprising a portion of video data and based on a NAL unit type of the first NAL unit, detect a broken link picture.

[0136] FIG. 5 is a block diagram illustrating an example set of devices that form part of network 100. In this example, network 10 includes routing devices 104A, 104B (routing devices 104) and transcoding device 106. Routing devices 104 and transcoding device 106 are intended to represent a small number of devices that may form part of network 100. Other network devices, such as switches, hubs, gateways, firewalls, bridges, and other such devices may also be included within network 100. Moreover, additional network devices may be provided along a network path between server device 102 and client device 108. Server device 102 may correspond to source device 12 (FIG. 1), while client device 108 may correspond to destination device 14 (FIG. 1), in some examples.

[0137] In general, routing devices 104 implement one or more routing protocols to exchange network data through network 100. In some examples, routing devices 104 may be configured to perform proxy or cache operations. Therefore, in some examples, routing devices 104 may be referred to as proxy devices. In general, routing devices 104 execute routing protocols to discover routes through network 100. By executing such routing protocols, routing device 104B may discover a network route from itself to server device 102 via routing device 104A.

[0138] The techniques of this disclosure may be implemented by network devices such routing devices 104 and transcoding device 106, but also may be implemented by client device 108. In this manner, routing devices 104, transcoding device 106, and client device 108 represent examples of devices configured to perform the techniques of this disclosure. Moreover, the devices of FIG. 1, and encoder illustrated in FIG. 3 and the decoder illustrated in FIG. 4 are also exemplary devices that can be configured to perform the techniques of this disclosure.

[0139] For examples, server device 102 may include an encoder to encode a picture or pictures that are after a random access point or stream adaptation point, such as a temporal layer switching point, or other stream adaptation point. For example, this point could be a switching point for adaptation of bit rate, frame rate (i.e., temporal layer switching point), or spatial resolution Similarly, client device 108 may decode a picture or pictures that are after a random access point or stream adaptation point, such as a temporal layer switching point. Again, this could be a switching point for adaptation of bit rate, frame rate (i.e., temporal layer switching point), or spatial resolution. One or more of these pictures may be leading pictures. The leading pictures cannot be correctly decoded at client device 108 when random access from the BLA picture occurs.

[0140] In an example, to prevent error propagation from reference pictures that may not be available depending on where the decoding starts, client device 108 may not apply pictures stored in the decoded picture buffer (DPB) that precedes the BLA picture either in decoding order or output order (which includes the leading pictures) as reference pictures.

[0141] In various examples, client device 108 may mark all reference pictures in the DPB as unused for reference before decoding the BLA picture. For example, client device 108 may mark reference pictures in the decode picture buffer (DPB) as unused for reference.

[0142] In another example, server device 102, client device 108, or both, may include a syntax element or flag in a slice header and encode the flag into a bitstream, e.g., a no_output_of_prior_pics_flag, so that none of the prior pictures in the DPB processed by the video decoder 30 to decode leading pictures are output from the DPB, e.g., for presentation on a display monitor. In some examples, this flag may be early in the slice header before entropy decoding so that it can be more easily decoded and the information is available earlier in the coding process. In an example, one of these network element devices may convert a CRA to a broken link picture when, for example, splicing or channel switching or stream adaptation, such as temporal layer switching, is needed. Having the flag may accessible without entropy coding allows for network elements without the ability to entropy decode to have access to the flag.

[0143] The plurality of NAL unit types includes one or more of (1) a coded slice of a broken link picture, wherein the broken link picture is a broken link picture with associated leading pictures (either decodable or non-decodable) in a bitstream, (2) a

coded slice of a broken link picture, wherein the broken link picture is a broken link picture with associated decodable leading pictures in the bitstream, and (3) a coded slice of a broken link picture, wherein the broken link picture is a broken link picture with no associated leading pictures in the bitstream. In an example, the leading picture comprises a picture that precedes a random access picture (RAP) in a display order but follows the random access picture in decoding order.

[0144] In various examples, routing devices 104A, 104B, and transcoding device 106 that make up network 100 may also perform some processing on a picture or pictures that are after a random access point or stream adaptation point, such as a temporal layer switching point. For example, this could be a switching point for adaptation of bit rate, frame rate (i.e., temporal layer switching point), or spatial resolution. As discussed above, one or more of these pictures may be leading pictures that cannot be correctly decoded.

[0145] In an example, one or more of routing devices 104A, 104B, and transcoding device 106 may not use any picture that precedes the CRA picture either in decoding order or output order (which includes the leading pictures) as reference pictures. In another example, one or more of routing devices 104A, 104B, and transcoding device 106 may mark all reference pictures in the DPB as unused for reference before decoding the BLA picture. For example, one or more of routing devices 104A, 104B, and transcoding device 106 may mark reference pictures in the decoded picture buffer (DPB) as unused for reference. In another example, one or more of routing devices 104A, 104B, and transcoding device 106 or a streaming server may use a flag, e.g., a no_output_of_prior_pics_flag so that none of the prior pictures in the DPB are output. Additionally, routing devices 104A, 104B, and transcoding device 106 may process NAL unit types assigned to indicate when BLA pictures have and do not have leading pictures.

[0146] FIG. 6 is a diagram illustrating an example in accordance with the techniques described in this disclosure. FIG. 6 illustrates examples of cases when leading pictures are decodable and non-decodable. The decodability of leading pictures may be based on location of the predictive block. Additionally, the decodability of leading pictures may be based on whether the current CRA picture is a CRA picture that is not a BLA picture or the current CRA picture is a CRA picture that is also a BLA picture. (BLA pictures are a subset of the CRA pictures.)

[0147] Portion 200 of FIG. 6 illustrates a series of pictures in decode order. Initially, video decoder 30 (FIGS. 1 and 4) or client device 108 (FIG. 5) for example, may decode a RAP picture at location 202 as a first picture in a video sequence. Video decoder 30 or client device 108 may then decode a picture or portion of a picture that may act as a reference picture at location 204. As illustrated in FIG. 6, location 204 is a possible location of the reference picture in decoding order. If the reference picture is located at location 204 and picture at location 206 is a CRA picture that is not a BLA picture, then the leading picture at location 208 will be decodable. Conversely, if the reference picture is located at location 204 and the picture at location 206 is a CRA picture that is also a broken link picture, then the leading picture at location 208 will not be decodable. (broken link pictures, referred to as BLA or BLC pictures, are a subset of the CRA pictures.)

[0148] If the RAP picture at location 202 is the RAP picture, where decoding starts and the current CRA picture at location 206 is not a broken link picture then, as described above, the leading picture at location 208 is decodable. Conversely, if current CRA picture at location 206 is the RAP, then the current CRA picture at location 206 is also a broken link picture and the leading picture at location 208 is not decodable. This is because the predictive block at location 204 is not available for a broken link picture that is a current CRA picture at location 206. Accordingly, the predictive block at location 204 may be (1) marked as unused for reference and (2) the no_output_of_prior_pics_flag may indicate that the prior pictures, up to and including the leading picture at location 208, should not be output.

[0149] As illustrated in FIG. 6, location 210 is another possible location of the reference picture in decode order. If the reference picture is located at location 210 then leading picture at location 208 will be decodable.

[0150] Portion 212 of FIG. 6 illustrates a series of pictures in output order. Initially, video decoder 30 (FIGS. 1 and 4) or client device 108 (FIG. 5) a RAP picture 202. Video decoder 30 or client device 108 may then decode a picture or portion of a picture that may act as a predictive block at location 204. As illustrated in FIG. 6, location 204 is a possible location of the predictive block in output order.

[0151] In output order, leading picture 208 may be output before current CRA picture at location 206, as illustrated in FIG. 6. As illustrated in FIG. 6, location 210 is another possible location of the predictive block in output order.

[0152] FIG. 7 is a flowchart illustrating an example method for coding RAP pictures in accordance with one or more examples described in this disclosure. As illustrated in FIG. 7, in some examples, a video coder, e.g., video encoder 20 or video decoder 30 may code a BLA picture for a random access point or stream adaptation point, such as a temporal layer switching point in a video sequence. For example, this could be a switching point for adaptation of bit rate, frame rate or spatial resolution. The BLA picture may include one or more leading pictures. The leading pictures cannot be correctly decoded (e.g., by decoder 30, a MANE, or other decoding device) when random access from BLA picture occurs.

[0153] In an example, to prevent error propagation from reference pictures that may not be available depending on where the decoding starts, video coder may not use any picture that precedes the BLA picture either in decoding order or output order (which includes the leading pictures) as reference pictures.

[0154] For example, video decoder 30 may receive and decode a syntax element that indicates that a picture storage buffer is emptied without outputting any pictures from the picture storage buffer (700). The syntax element may be included in the encoded video bitstream by encoder 20 or an intermediate network element. Video decoder 30 may determine whether the syntax element indicates that a no output of prior pictures should occur; for example, video decoder 30 may check the syntax element (702) to determine whether it is set, i.e., equal to 1. When the video decoder 30 receives a syntax element that is set, the video decoder 30 may cause pictures prior to a current picture in decoding order and residing in the picture storage buffer at the time of the decoding of the current picture to be emptied from the picture storage buffer without being output (704).

[0155] In some examples, the syntax element may be one of a plurality of syntax elements. Additionally, the plurality of syntax elements may include entropy coded slice header syntax elements and non-entropy coded slice header syntax elements. In an example, the syntax element may indicate that a picture storage buffer is empty without outputting any pictures from the decoded picture buffer is included in the slice header before any entropy coded slice header syntax element. In another example, the syntax element may indicate that data in the picture storage buffer should be ignored and/or overwritten without outputting any pictures from the picture storage buffer is included in the slice header before any entropy coded slice header syntax element. In some examples, the syntax element may be a no_output_of_prior_pics_flag and the

no_output_of_prior_pics_flag may be included in the slice header immediately after a first_slice_in_pic_flag. The first_slice_in_pic_flag may be a flag that indicates whether a slice is the first slice, in decoding order, of a picture.

[0156] FIG. 8 is a flowchart illustrating an example method in accordance with one or more examples described in this disclosure. In the illustrated example, a video coder may determine that the current picture is a CRA picture (800). The video coder may also determine that the current picture is a RAP picture (802). The video coder may determine that the current picture is a BLA picture when the current picture is both a CRA picture and a RAP picture (804). In some examples, a video coder may use a NAL unit type to determine if that the current picture is a BLA picture. The video coder may mark reference pictures as unused for reference (806).

[0157] In various examples, the video coder may mark all reference pictures in the DPB as unused for reference before decoding the BLA picture. For example, video decoder 30 may mark reference pictures in the picture memory 92, sometimes referred to as the decoded picture buffer (DPB), as unused for reference. Accordingly, the pictures will not be used for inter-coding, which will avoid possible errors, and in some examples may address issues with spatial adaptation. Additionally, generally these pictures will not be output to, for example, a monitor or screen, where they may be viewed.

[0158] Video decoder 30 may determine that a current picture is a broken link picture and marking a reference picture in a picture storage buffer as unused for reference prior to decoding the BLA picture. In an example, marking of the reference picture in the picture storage buffer may occur when the broken link picture comprises a non-decodable broken link picture, which may be determined in some examples, based on NAL unit type. In some examples, one or more of (1) marking pictures in a DPB as unused for reference, (2) using a syntax element such as a no_output_of_prior_pics_flag, and (3) using NAL unit types that indicate types of broken link pictures may be used, independently or in any combination.

[0159] FIG. 9 is a flowchart illustrating an example method in accordance with one or more examples described in this disclosure. The techniques of FIG. 9 may be performed by a video processing device, such as video decoder 30, network entity 29, routing device 104, transcoding device 106 or some other such video processing device. A video processing device may process NAL units where NAL unit types for the NAL units are assigned to indicate when BLA pictures have and do not have leading pictures. In some instances, the video processing device may, for example, detect that a picture is

a CRA picture and determine if the CRA picture is a BLA picture with no leading pictures, a BLA picture with decodable leading pictures, or a BLA picture with non-decodable leading pictures. Based on the determination, the video processing device may set a NAL unit type for the video data to indicate that the NAL unit includes a BLA picture with no leading pictures, a BLA picture with decodable leading pictures, or a BLA picture with non-decodable leading pictures.

[0160] In one example, the video processing device may code pictures according to one of a plurality of different network abstraction layer (NAL) unit types including one or more of the following. The video processing device may determine that a BLA picture has no associated leading pictures (900) and set a NAL unit to a NAL unit type that indicates that the NAL unit comprises the BLA picture without associated leading pictures (902). The video processing device may, for example, code a coded slice of the BLA picture or the entire BLA picture using a NAL unit type that indicates that the BLA picture is a BLA picture with no associated leading pictures in a bitstream. The video processing device may determine that a BLA picture has associated decodable leading pictures (904) and set a NAL unit to a NAL unit type that indicates that the NAL unit comprises the BLA picture with associated decodable leading pictures (906). The video processing device may, for example, code a coded slice or the entire BLA picture using a NAL unit type that indicates that the BLA picture is a BLA picture with associated decodable leading pictures in the bitstream. The video processing device may determine that a BLA picture has associated leading pictures (908) and set a NAL unit to a NAL unit type that indicates that the NAL unit comprises the BLA picture with associated leading pictures (910). The video processing device may, for example, code a coded slice of the BLA picture or the entire BLA picture using a NAL unit type that indicates that the BLA picture is a BLA picture with associated leading pictures in the bitstream. In an example, if video decoder 30 detects a BLA NAL unit type, then video decoder 30 may, for example, mark pictures in the DPB as unused for reference.

[0161] In some examples, one or more of (1) marking pictures in a DPB as unused for reference, (2) using a syntax element such as a no_output_of_prior_pics_flag, and (3) using NAL unit types that indicate BLA pictures may be used, independently or in any combination. For example, in some cases all three of these may be used. In other examples, the example NAL unit types might be used in conjunction with marking pictures in the DPB as unused for reference. In another example, the no output of prior pictures syntax element and marking the pictures as unused for reference might be used.

In another example, the no output of prior pictures syntax elements and NAL unit types might be used.

[0162] As used herein, BLC refers to Broken-Link Clean random access, BLCL refers to a BLC picture with associated leading pictures in the bitstream and BLCNL refers to a BLC picture with no associated leading pictures in the bitstream. As discussed herein, BLC pictures are generally the same as BLA pictures. CRA refers to Clean Random Access, CRAL refers to a CRA picture with associated leading pictures in the bitstream, and CRANL refers to a CRA picture with no associated leading pictures in the bitstream. IDR refers to Instantaneous Decoding Refresh, LPR refers to leading picture associated with a random access point picture, NSP refers to Nothing Special Plain, RAP refers to Random Access Point, and RPS refers to Reference Picture Set. As used herein TLA refers to Temporal Layer Access, TLAL refers to TLA picture that is also an LRP picture, TLANL refers to TLA picture that is not an LRP picture.

[0163] BLA access unit refers to an access unit in which the coded picture is a BLA picture. A BLC picture is a RAP picture in which the slice header of the coded slices includes RPS syntax while the RPS is derived as empty without using the RPS syntax. A BLCL access unit is an access unit in which the coded picture is a BLCL picture. BLCL pictures are BLA pictures for which the associated LPR pictures are present in the bitstream. In some examples, a BLCL access unit may be equivalent to a combination of BLA_W_DLP and BLA_W_LP. A BLCNL access unit is an access unit in which the coded picture is a BLCNL picture. In some examples, a BLCNL access unit may be equivalent to BLA_N_LP. BLCNL pictures are BLA pictures for which the associated LPR pictures are not present in the bitstream.

[0164] In an example, a CRA access unit is an access unit in which the coded picture is a CRA picture. CRA pictures are RAP picture in which the slice header of the coded slices includes RPS syntax and the RPS syntax is used for derivation of the RPS. A CRAL access unit is an access unit in which the coded picture is a CRAL picture. CRAL pictures are CRA picture for which the associated LPR pictures are present in the bitstream. A CRANL access unit is an access unit in which the coded picture is a CRANL picture. CRANL pictures are CRA pictures for which the associated LPR pictures are not present in the bitstream.

[0165] In an example, an IDR access unit is an access unit in which the coded picture is an IDR picture. IDR pictures are RAP pictures in which the slice header of the coded slices does not include RPS syntax and the RPS is derived as empty.

[0166] In an example, if all preceding access units in decoding order are not present, provided that each parameter set referred to by the coded picture and all subsequent coded pictures in decoding order is present before its activation, an IDR picture and all subsequent coded pictures in decoding order can be correctly decoded. Alternatively, in another example, IDR pictures may be as defined in HEVC with the addition of the preceding.

[0167] In other examples, an IDR picture may be a coded picture containing only I slices. Additionally, for an example IDR picture, all coded pictures that follow the IDR picture in decoding order do not use inter prediction from any picture that precedes the IDR picture in decoding order. In an example, any picture that precedes the IDR picture in decoding order also precedes the IDR picture in output order.

[0168] A leading picture is a coded picture that is not a RAP picture and that follows some other particular picture in decoding order and precedes the particular picture in output order. An LPR picture is a leading picture that is associated with a RAP picture or a leading picture of a RAP picture.

[0169] Picture order count may be a variable that is associated with each coded picture and has a value that is increasing with increasing picture position in output order relative to the previous RAP picture in decoding order.

[0170] In an example, a RAP access unit is an access unit in which the coded picture is a RAP picture. A RAP picture may be a coded picture containing only I slices. For a RAP picture, all coded pictures that follow the RAP picture both in decoding order and output order do not use inter prediction from any picture that precedes the RAP picture either in decoding order or output order. The output of any picture that precedes the RAP picture in decoding order shall precede the output of the RAP picture. If all preceding access units in decoding order are not present, provided that each parameter set referred to by the coded picture and all subsequent coded pictures in decoding order is present before its activation, a RAP picture and all subsequent coded pictures in both decoding order and output order can be correctly decoded.

[0171] Alternatively, RAP picture may be defined consistent with the preceding discuss and as follows. A RAP picture may be a coded picture containing only I slices and for which all coded pictures that follow the RAP picture both in decoding order and output order do not use inter prediction from any picture that precedes the RAP picture either in decoding order or output order. Any picture that precedes the RAP picture in decoding order also precedes the RAP picture in output order.

[0172] A TLA access unit is an access unit in which the coded picture is a TLA picture. A TLA picture is a coded picture for which the TLA picture and all coded pictures with temporal_id greater than or equal to the temporal_id of the TLA picture. A TLA picture that follows the TLA picture in decoding order shall not use inter prediction from any picture with temporal_id greater than or equal to the temporal_id of a TLA picture that precedes the TLA picture in decoding order. A TLAL access unit is an access unit in which the coded picture is a TLA picture.

[0173] In some examples, the following distinct VCL NAL unit types may be defined. As a first example, a NAL unit type may be provided for a coded slice of an IDR picture (e.g., nal_unit_type = 5). For this NAL unit type, the IDR picture concept in HEVC WD6 applies. A unique feature of this VCL NAL unit type compared to other types of VCL NAL units is that there is no reference picture set (RPS) syntax included in the slice header.

[0174] Some examples include a coded slice of a BLCNL picture (a BLC picture with no associated leading pictures in the bitstream, e.g., nal_unit_type = 2). Compared to a coded slice of an IDR picture, a coded slice of a BLCNL picture includes RPS syntax in the slice header, but the RPS syntax is not used for RPS derivation, but rather all RPS subsets are derived to be empty.

[0175] Some examples include a coded slice of a BLCL picture (a BLC picture with associated leading pictures in the bitstream, e.g., nal_unit_type = 3) Compared to a BLCNL picture, there are leading pictures associated with a BLCL picture in the bitstream.

[0176] Some examples include a coded slice of a CRANL picture (a CRA with no associated leading pictures in the bitstream, e.g., nal_unit_type = 15). Compared to a coded slice of a BLCNL picture, a coded slice of a CRANL picture includes RPS syntax in the slice header, and the RPS syntax is used for RPS derivation.

[0177] Some examples include a coded slice of a CRAL picture (a CRA picture with associated leading pictures in the bitstream, e.g., nal_unit_type = 4). Compared to a CRANL picture, there are leading pictures associated with a CRAL picture in the bitstream.

[0178] Some examples include a coded slice of a TLANL picture (a TLA picture that is not an LFR picture, e.g., nal_unit_type = 16). Some examples include a coded slice of a TLAL picture (a TLA picture that is also an LPR picture, e.g., nal_unit_type = 17).

Some examples include a coded slice of a NSP picture (a nothing-special plain picture - not any of the above, nal_unit_type = 1).

[0179] Before decoding the slice data of each BLC picture (BLCL or BLCNL) picture, all reference pictures in the decoded picture buffer (DPB) must be marked as "unused for reference" by decoder 30, as described above. Only with this in place, the current decoding process as currently specified in HEVC WD 6 for leading pictures of a CRA picture that starts the bitstream can be directly applied by decoder 30 for leading pictures of a BLA picture, regardless of whether the BLA picture changes the spatial resolution.

[0180] Without the above, if a BLA picture does not change the spatial resolution, the current decoding process as currently specified in HEVC WD 6 for leading pictures of a CRA picture that starts the bitstream can be directly applied by decoder 30 for leading pictures of the BLA picture. However if a BLA picture changes the spatial resolution, then the current decoding process as currently specified in HEVC WD 6 for leading pictures of a CRA picture that starts the bitstream cannot be directly applied for leading pictures of the BLA picture, as the situation may appear that the spatial resolution is different for a current picture and a reference picture for the current picture.

[0181] One way to ensure that all reference pictures in the DPB are to be marked as "unused for reference" before decoding the slice data of each BLA picture is to derive the RPS of each BLA picture to be empty, regardless of whether the RPS signaling in the slice header indicates non-empty RPS. For example, even if there is an RPS, the video decoder 30 may override this and derives or treats the RPS as empty if the picture is a BLA picture.

[0182] Actually, if the RPS signaling in the slice header does indicate empty RPS for a BLA picture or a CRA (CRAL or CRANL) picture, then the picture should have been coded as an IDR picture.

[0183] In some examples, a RAP picture may be defined as a coded picture containing only I slices. For a RAP picture, all coded pictures that follow the RAP picture both in decoding order and output order do not use inter prediction from any picture that precedes the RAP picture either in decoding order or output order. Additionally, the output of any picture that precedes the RAP picture in decoding order may precede the output of the RAP picture.

[0184] To ensure that the output of any picture that precedes the RAP picture in decoding order shall precede the output of the RAP picture, one way is for a video

encoder 20 to set the no_output_of_prior_pics_flag equal to 1 in a coded bitstream transmitted to, for example, a video decoder 30, as discussed above. In another example, a video decoder 30 may infer no_output_of_prior_pics_flag to be equal to 1 (regardless of its value) for a BLA picture. This way, splicing operations at a BLA picture, wherein in the spliced bitstream the POC value of a picture earlier than the BLA picture is greater than the POC value of the BLA picture, is allowed. In particular, if the POC value of the BLA picture is derived as equal to its POC LSB (by assuming POC MSB equal to 0), the above can easily happen. Another way to ensure this is to make sure that the output times of pictures that precede the BLA picture in decoding order are earlier than that of the BLA picture.

[0185] Some examples allow the video encoder 20 and/or a bitstream splicer to determine which of the ways described herein to use. Accordingly, video encoder 20, for example, may include a no_output_of_prior_pics_flag in the slice header of BLA pictures in some examples. In other examples, video encoder 20 may include NAL unit types that indicate if pictures stored in a buffer at video decoder 30 may be used for reference. Accordingly, video decoder 30 may mark pictures as unused for reference, for example, when the use of such pictures in decoding will result in incorrectly decoding a picture.

[0186] To enable simple rewriting of a CRA to a BLA picture by a network element, in addition to putting the no_output_of_prior_pics_flag in the slice header of a CRA picture, the no_output_of_prior_pics_flag may also be included as early as possible in the slice header. This may be before any entropy coded slice header parameters, e.g., immediately after first_slice_in_pic_flag in some examples, as discussed above.

[0187] In an example, two back-to-back BLA pictures have the same POC LSB, and it is only possible to differentiate them by random_access_pic_id (or renamed to rap_pic_id). Thus, it might be preferable to use fixed-length coding for rap_pic_id, and put rap_pic_id relatively early in the slice header, preferably not after any entropy coded sliced header parameters, e.g., immediately after first_slice_in_pic_flag and no_output_of_prior_pics_flag, e.g., for both CRA pictures and BLA pictures. Other slice header syntax elements that may be used for picture boundary detection, e.g., pic_parameter_set_id, and POC LSB (i.e., pic_order_cnt_lsb) may be similar.

[0188] All parameter sets of a particular type (e.g., SPS) or all types may be present in the beginning of the bitstream, i.e., included in the first access unit in the bitstream. If so, it is convenient to fetch all the parameter sets of a particular type and send them out-

of-band. For example, a coder may include in a Session Description Protocol (SDP) parameter which is used during session negotiation. Therefore, it may be beneficial for an encoder to include an indication in the bitstream that all parameter sets of a particular type or all types are present in the beginning of the bitstream. The indication may be included in an SEI message, an access unit delimiter, or a parameter set. For example, a distinct NAL unit type may be used for SPS, picture parameter set (PPS), or adaptation parameter set (APS) to indicate that all SPSs (or PPSs or APSs) are present in the beginning of the bitstream.

[0189] The syntax element slice_type may be conditionally present in the slice header, based on whether the NAL unit type indicates that the picture containing the slice is an IDR picture, a CRA picture, or a BLA picture. For example, if The NAL unit type indicates that the picture containing the slice is an IDR picture, a CRA picture, or a BLA picture, slice_type is not present in the slice header. Otherwise, the encoder inserts the slice_type in the slice header. When not present, the value of slice_type indicates that the slice is an I slice.

[0190] FIG. 10 is a flowchart illustrating an example method for decoding a slice in accordance with one or more examples described in this disclosure. In one example, video decoder 30 may decode video data, including receiving a slice of a current picture to be decoded for a sequence of video data (1000). Video decoder 30 may receive, in a slice header of the slice, at least one entropy coded syntax element and at least one non-entropy coded syntax element. The non-entropy coded syntax element may be before the entropy coded syntax element in the slice header. Additionally, the non-entropy coded syntax element may indicate whether pictures prior to the current picture in decoding order are to be emptied from a picture storage buffer without being output (1002). In one example, the syntax element may be a no_output_of_prior_pics_flag. The no_output_of_prior_pics_flag may be set to "1," e.g., to indicate when pictures prior to the current picture in decoding order are to be emptied from a picture storage buffer without being output. Video decoder 30 may decode the slice based on the non-entropy coded syntax element (1004).

[0191] FIG. 11 is a flowchart illustrating an example method for encoding a slice in accordance with one or more examples described in this disclosure. Encoder 20 may encode video data. Video encoder 20 may encode a slice of a current picture for a sequence of video data (1100).

[0192] Video encoder 20 encodes, in a slice header of the slice, at least one entropy encoded syntax element and at least one non-entropy coded syntax element, wherein the non-entropy encoded syntax element is before the entropy encoded syntax element in the slice header and indicates whether pictures prior to the current picture in decoding order are to be emptied from a picture storage buffer without being output (1102). In one example, the syntax element may be a no_output_of_prior_pics_flag. The no_output_of_prior_pics_flag may be set to "1," e.g., to indicate when pictures prior to the current picture in decoding order are to be emptied from a picture storage buffer without being output.

[0193] FIG. 12 is a flowchart illustrating an example technique for processing video data, as described in this disclosure. The techniques of FIG. 12 may be performed by a video processing device, such as video decoder 30, network entity 29, routing device 104, transcoding device 106 or some other such video processing device. The video processing device can receive in the video data a NAL unit (1210), and based on a NAL unit type of the first NAL unit, detect a broken link picture (1220). The NAL unit type may be a NAL unit type indicating the broken link picture comprises a broken link picture with leading pictures, or the NAL unit type may be a different NAL unit type indicating the broken link picture comprises a broken link picture without leading pictures. The video processing entity may detect the broken link picture without performing entropy decoding of syntax elements of video bitstream data in the NAL unit.

[0194] Based on the type of NAL unit detected, the video processing device may perform one or more of a variety of operations. A MANE, for example, may make one or more routing decisions for the NAL unit based on the NAL unit type. As one example, if the video processing device is a MANE, the video processing device may, in response to a NAL unit indicating a broken link picture with leading pictures, discard the leading pictures and transmit the NAL unit without the leading pictures. In instances where the NAL unit indicates a broken link picture without leading pictures, the video processing device may transmit the NAL unit without having to discard any leading pictures. The MANE may, for example, transmit the NAL unit to a video decoder, potentially via one or more routing devices.

[0195] If, for example, the video processing device is a video decoder as opposed to a MANE, then the video decoder may perform one or more of a variety of different

operations. For example, in response to detecting a broken link picture, the video decoder may activating one or more new parameter sets for video data.

[0196] In one or more examples, the functions described may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the functions may be stored on or transmitted over, as one or more instructions or code, a computer-readable medium and executed by a hardware-based processing unit. Computer-readable media may include computer-readable storage media, which corresponds to a tangible medium such as data storage media, or communication media including any medium that facilitates transfer of a computer program from one place to another, e.g., according to a communication protocol. In this manner, computer-readable media generally may correspond to (1) tangible computer-readable storage media which is non-transitory or (2) a communication medium such as a signal or carrier wave. Data storage media may be any available media that can be accessed by one or more computers or one or more processors to retrieve instructions, code and/or data structures for implementation of the techniques described in this disclosure. A computer program product may include a computer-readable medium.

[0197] In still other examples, this disclosure contemplates a computer readable medium comprising a data structure stored thereon, wherein the data structure includes an encoded bitstream consistent with this disclosure. In particular, the data structures may include the NAL unit designs described herein.

[0198] By way of example, and not limitation, such computer-readable storage media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage, or other magnetic storage devices, flash memory, or any other medium that can be used to store desired program code in the form of instructions or data structures and that can be accessed by a computer. Also, any connection is properly termed a computer-readable medium. For example, if instructions are transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, DSL, or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. It should be understood, however, that computer-readable storage media and data storage media do not include connections, carrier waves, signals, or other transient media, but are instead directed to non-transient, tangible storage media. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc

(DVD), floppy disk and Blu-ray disc, where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above should also be included within the scope of computer-readable media.

[0199] Instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated hardware and/or software modules configured for encoding and decoding, or incorporated in a combined codec. Also, the techniques could be fully implemented in one or more circuits or logic elements.

[0200] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, an integrated circuit (IC) or a set of ICs (e.g., a chip set). Various components, modules, or units are described in this disclosure to emphasize functional aspects of devices configured to perform the disclosed techniques, but do not necessarily require realization by different hardware units. Rather, as described above, various units may be combined in a codec hardware unit or provided by a collection of interoperative hardware units, including one or more processors as described above, in conjunction with suitable software and/or firmware.

[0201] Various examples have been described. These and other examples are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A method for processing video data, the method comprising:
receiving a first network abstraction layer (NAL) unit comprising a portion of the video data; and
based on a NAL unit type of the first NAL unit, detecting a broken link picture.

2. The method of claim 1, wherein the NAL unit type is a first NAL unit type, and detecting the broken link picture comprises detecting a broken link picture with leading pictures.

3. The method of claim 2, further comprising:
discarding the leading the pictures; and
transmitting the first NAL unit to a video processing device.

4. The method of claim 1, wherein the NAL unit type is a first NAL unit type, and detecting the broken link picture comprises detecting a broken link picture without leading pictures.

5. The method of claim 4, further comprising:
transmitting the first NAL unit to a video processing device.

6. The method of claim 4 further comprising:
receiving a second NAL unit;
determining a NAL unit type for the second NAL unit; and
based on the NAL unit type for the second NAL unit, detecting a broken link picture with leading pictures, wherein the NAL unit type for the second NAL unit is different than the first NAL unit type.

7. The method of claim 1, wherein the NAL unit type is a first NAL unit type, and detecting the broken link picture comprises detecting a broken link picture with decodable leading pictures.

8. The method of claim 1, wherein the method is performed by a video processing entity, and wherein the video processing entity detects the broken link picture without performing entropy decoding of syntax elements of video bitstream data in the NAL unit.

9. The method of claim 1, further comprising:
in response to detecting a broken link picture, activating one or more new parameter sets for coding the video data.

10. The method of claim 9, wherein the one or more new parameter sets comprise one or more of an adaptation parameter set (APS), a sequence parameter set (SPS), and a picture parameter set (PPS).

11. The method of claim 9, wherein an identification (ID) for one of the one or more parameter sets is the same as an active parameter set for a previously decoded picture.

12. The method of claim 1, wherein the method is performed by a video decoder.

13. The method of claim 1, wherein the method is performed by a media aware network element (MANE).

14. A method for processing video data, the method comprising:
receiving a network abstraction layer (NAL) unit comprising a portion of the video data;
determining that the NAL unit comprises a broken link picture; and
setting a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

15. The method of claim 14, wherein determining that the NAL unit comprises the broken link picture comprises determining that the NAL unit comprises a broken link picture with leading pictures and wherein the NAL unit type indicates that the NAL unit comprises the broken link picture with leading pictures.

16. The method of claim 15, wherein determining that the NAL unit comprises the broken link picture with leading pictures comprises determining that the NAL unit comprises a broken link picture with decodable leading pictures and wherein the first NAL unit type indicates the NAL unit comprises the broken link picture with decodable leading pictures.

17. The method of claim 15, wherein determining the NAL unit comprises the broken link picture with leading pictures comprises determining the NAL unit comprises a broken link picture with non-decodable leading pictures and wherein the first NAL unit type indicates the NAL unit comprises the broken link picture with leading pictures that are decodable or non-decodable.

18. The method of claim 14, wherein determining that the NAL unit comprises the broken link picture comprises determining that the NAL unit comprises a broken link picture without leading pictures and wherein the first NAL unit type indicates the NAL unit comprises the broken link picture without leading pictures.

19. A device for processing video data, the device comprising one or more processors configured to:

receive a first network abstraction layer (NAL) unit comprising a portion of the video data; and

based on a NAL unit type of the first NAL unit, detect a broken link picture.

20. The device of claim 19, wherein the NAL unit type is a first NAL unit type, and detecting the broken link picture comprises detecting a broken link picture with leading pictures.

21. The device of claim 20, wherein the one or more processors are further configured to:

discard the leading the pictures; and

transmit the first NAL unit to a video processing device.

22. The device of claim 19, wherein the NAL unit type is a first NAL unit type, and detecting the broken link picture comprises detecting a broken link picture without leading pictures.

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

transmit the first NAL unit to a video processing device.

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

receive a second NAL unit;

determine a NAL unit type for the second NAL unit; and

based on the NAL unit type for the second NAL unit, detect a broken link picture with leading pictures, wherein the NAL unit type for the second NAL unit is different than the first NAL unit type.

25. The device of claim 19, wherein the NAL unit type is a first NAL unit type, and detecting the broken link picture comprises detecting a broken link picture with decodable leading pictures.

26. The device of claim 19, wherein the device comprises a video processing entity, and wherein the video processing entity is configured to detect the broken link picture without performing entropy decoding of syntax elements of video bitstream data in the NAL unit.

27. The device of claim 19, wherein the one or more processors are further configured to:

in response to detecting a broken link picture, activate one or more new parameter sets for coding the video data.

28. The device of claim 27, wherein the one or more new parameter sets comprise one or more of an adaptation parameter set (APS), a sequence parameter set (SPS), and a picture parameter set (PPS).

29. The device of claim 27, wherein an identification (ID) for one of the one or more parameter sets is the same as an active parameter set for a previously decoded picture.

30. The device of claim 19, wherein the device comprises a video decoder.

31. The device of claim 19, wherein the device comprises a media aware network element (MANE).

32. The device of claim 19, wherein the device comprises at least one of:
an integrated circuit;
a microprocessor; and,
a wireless communication device that comprises a video decoder.

33. A device for processing video data, the device comprising one or more processors configured to:
receive a network abstraction layer (NAL) unit comprising a portion of the video data;
determine that the NAL unit comprises a broken link picture; and
set a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

34. The device of claim 33, wherein the one or more processors are configured to determine that the NAL unit comprises the broken link picture by determining that the NAL unit comprises a broken link picture with leading pictures and wherein the NAL unit type indicates that the NAL unit comprises the broken link picture with leading pictures.

35. The device of claim 34, wherein the one or more processors are configured to determine that the NAL unit comprises the broken link picture with leading pictures by determining that the NAL unit comprises a broken link picture with decodable leading pictures and wherein the first NAL unit type indicates the NAL unit comprises the broken link picture with decodable leading pictures.

36. The device of claim 34, wherein the one or more processors are configured to determine the NAL unit comprises the broken link picture with leading pictures by determining the NAL unit comprises a broken link picture with non-decodable leading pictures and wherein the first NAL unit type indicates the NAL unit comprises the broken link picture with leading pictures that are decodable or non-decodable.

37. The device of claim 33, wherein the one or more processors are configured to determine that the NAL unit comprises the broken link picture by determining that the NAL unit comprises a broken link picture without leading pictures and wherein the first NAL unit type indicates the NAL unit comprises the broken link picture without leading pictures.

38. A device for processing video data, the device comprising:
means for receiving a first network abstraction layer (NAL) unit comprising a portion of the video data; and
means for detecting a broken link picture based on a NAL unit type of the first NAL unit.

39. A device for processing video data, the device comprising:
means for receiving a network abstraction layer (NAL) unit comprising a portion of the video data;
means for determining that the NAL unit comprises a broken link picture; and
means for setting a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.

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

receive a first network abstraction layer (NAL) unit comprising a portion of the video data; and

based on a NAL unit type of the first NAL unit, detect a broken link picture.

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

receive a network abstraction layer (NAL) unit comprising a portion of the video data;

determine that the NAL unit comprises a broken link picture; and

set a NAL unit type for the NAL unit to a NAL unit type that indicates that the NAL unit comprises the broken link picture.