

A METHOD FOR ANNOTATING VVC SUBPICTURES IN DASH

CROSS REFERENCE TO RELATED APPLICATION

[1] The present application claims priority to provisional application US 63/298,924, filed on January 12, 2022, and application US 18/073,984, filed on December 2, 2022, the contents of which are hereby expressly incorporated by reference, in their entirety, into the present application.

BACKGROUND

1. Field

[2] The present disclosure is directed to dynamic adaptive streaming over HTTP (DASH) signaling.

2. Description of Related Art

[3] MPEG DASH provides a standard for streaming multimedia content over IP networks. ISO/IEC 23009-1 DASH standard allows the streaming of the multi-rate content. The DASH manifest, MPD, can describe various media content. While the DASH standard provides a way to describe various content and their relation, it does not provide an interoperable solution to annotate the VVC subpictures to be used for picture-in-picture applications.

SUMMARY

[4] To address one or more different technical problems, this disclosure provides technical solutions to reduce network overhead and server computational overheads, and there is included a method and apparatus comprising memory configured to store computer program code and a processor or processors configured to access the computer program code and operate as

instructed by the computer program code. The computer program code comprises obtaining code configured to cause the at least one processor to obtain a dynamic adaptive streaming over HTTP (DASH) video bitstream comprising one or more subpictures, determining code configured to cause the at least one processor to determine whether the one or more subpictures include Versatile Video Coding (VVC) compliant subpictures, annotating code configured to cause the at least one processor to, in response to the DASH video bitstream including the VVC compliant subpictures, annotate the one or more subpictures based on one or more flags, and manipulating code configured to cause the at least one processor to manipulate the DASH video stream based on the annotated one or more subpictures.

[5] According to exemplary embodiments, the video data is in dynamic adaptive streaming over HTTP (DASH), and the client is a DASH client.

[6] According to exemplary embodiments, determining whether to annotate the one or more of the first subpictures and the second subpictures comprises determining whether to annotate the one or more of the first subpictures and the second subpictures in a DASH media picture description (MPD).

[7] According to exemplary embodiments, the other of the first stream and the second stream comprises a picture-in-picture (PIP) stream.

[8] According to exemplary embodiments, controlling the client to replace the at least part of the one of the first stream and the second stream comprises replacing an area represented by a plurality of ones of the first subpictures and the second subpictures with the PIP stream.

[9] According to exemplary embodiments, the one of the first stream and the second stream represents a main video and the other of the first stream and the second stream represents a separately coded stream as a supplementary video to the main video.

[10] According to exemplary embodiments, controlling the client to replace the at least part of the one of the first stream and the second stream comprises determining whether a user requested to view the supplementary video.

[11] According to exemplary embodiments, the main video and the supplementary video each comprise versatile video coding (VVC) subpictures.

[12] According to exemplary embodiments, controlling the client to replace the at least part of the one of the first stream and the second stream comprises merging properties of versatile video coding (VVC) subpictures such that a decoder is provided with a single merged stream merged from both of the first stream and the second stream.

BRIEF DESCRIPTION OF THE DRAWINGS

[13] Further features, nature, and various advantages of the disclosed subject matter will be more apparent from the following detailed description and the accompanying drawings in which:

[14] Fig. 1 is a simplified schematic illustration in accordance with embodiments.

[15] Fig. 2 is a simplified schematic illustration in accordance with embodiments.

[16] Fig. 3 is a simplified block diagram regarding decoders in accordance with embodiments.

[17] Fig. 4 is a simplified block diagram regarding encoders in accordance with embodiments.

[18] Fig. 5 is a simplified block diagram in accordance with embodiments.

[19] Fig. 6 is a simplified diagram in accordance with embodiments.

[20] Fig. 7 is a simplified diagram in accordance with embodiments.

[21] Fig. 8 is a simplified flow diagram in accordance with embodiments.

[22] Fig. 9 is a schematic illustration in accordance with embodiments.

DETAILED DESCRIPTION

[23] The proposed features discussed below may be used separately or combined in any order. Further, the embodiments may be implemented by processing circuitry (e.g., one or more processors or one or more integrated circuits). In one example, the one or more processors execute a program that is stored in a non-transitory computer-readable medium.

[24] Fig. 1 illustrates a simplified block diagram of a communication system 100 according to an embodiment of the present disclosure. The communication system 100 may include at least two terminals 102 and 103 interconnected via a network 105. For unidirectional transmission of data, a first terminal 103 may code video data at a local location for transmission to the other terminal 102 via the network 105. The second terminal 102 may receive the coded video data of the other terminal from the network 105, decode the coded data and display the recovered video data. Unidirectional data transmission may be common in media serving applications and the like.

[25] Fig. 1 illustrates a second pair of terminals 101 and 104 provided to support bidirectional transmission of coded video that may occur, for example, during videoconferencing. For bidirectional transmission of data, each terminal 101 and 104 may code video data captured at a local location for transmission to the other terminal via the network 105. Each terminal 101 and 104 also may receive the coded video data transmitted by the other terminal, may decode the coded data and may display the recovered video data at a local display device.

[26] In Fig. 1, the terminals 101, 102, 103 and 104 may be illustrated as servers, personal computers and smart phones but the principles of the present disclosure are not so limited. Embodiments of the present disclosure find application with laptop computers, tablet computers, media players and/or dedicated video conferencing equipment. The network 105 represents any

number of networks that convey coded video data among the terminals 101, 102, 103 and 104, including for example wireline and/or wireless communication networks. The communication network 105 may exchange data in circuit-switched and/or packet-switched channels.

Representative networks include telecommunications networks, local area networks, wide area networks and/or the Internet. For the purposes of the present discussion, the architecture and topology of the network 105 may be immaterial to the operation of the present disclosure unless explained herein below.

[27] Fig. 2 illustrates, as an example for an application for the disclosed subject matter, the placement of a video encoder and decoder in a streaming environment. The disclosed subject matter can be equally applicable to other video enabled applications, including, for example, video conferencing, digital TV, storing of compressed video on digital media including CD, DVD, memory stick and the like, and so on.

[28] A streaming system may include a capture subsystem 203, that can include a video source 201, for example a digital camera, creating, for example, an uncompressed video sample stream 213. That sample stream 213 may be emphasized as a high data volume when compared to encoded video bitstreams and can be processed by an encoder 202 coupled to the camera 201. The encoder 202 can include hardware, software, or a combination thereof to enable or implement aspects of the disclosed subject matter as described in more detail below. The encoded video bitstream 204, which may be emphasized as a lower data volume when compared to the sample stream, can be stored on a streaming server 205 for future use. One or more streaming clients 212 and 207 can access the streaming server 205 to retrieve copies 208 and 206 of the encoded video bitstream 204. A client 212 can include a video decoder 211 which decodes the incoming copy of the encoded video bitstream 208 and creates an outgoing video

sample stream 210 that can be rendered on a display 209 or other rendering device (not depicted). In some streaming systems, the video bitstreams 204, 206 and 208 can be encoded according to certain video coding/compression standards. Examples of those standards are noted above and described further herein.

[29] Fig. 3 may be a functional block diagram of a video decoder 300 according to an embodiment of the present invention.

[30] A receiver 302 may receive one or more codec video sequences to be decoded by the decoder 300; in the same or another embodiment, one coded video sequence at a time, where the decoding of each coded video sequence is independent from other coded video sequences. The coded video sequence may be received from a channel 301, which may be a hardware/software link to a storage device which stores the encoded video data. The receiver 302 may receive the encoded video data with other data, for example, coded audio data and/or ancillary data streams, that may be forwarded to their respective using entities (not depicted). The receiver 302 may separate the coded video sequence from the other data. To combat network jitter, a buffer memory 303 may be coupled in between receiver 302 and entropy decoder / parser 304 (“parser” henceforth). When receiver 302 is receiving data from a store/forward device of sufficient bandwidth and controllability, or from an isosynchronous network, the buffer 303 may not be needed, or can be small. For use on best effort packet networks such as the Internet, the buffer 303 may be required, can be comparatively large and can advantageously of adaptive size.

[31] The video decoder 300 may include a parser 304 to reconstruct symbols 313 from the entropy coded video sequence. Categories of those symbols include information used to manage operation of the decoder 300, and potentially information to control a rendering device such as a display 312 that is not an integral part of the decoder but can be coupled to it. The control

information for the rendering device(s) may be in the form of Supplementary Enhancement Information (SEI messages) or Video Usability Information parameter set fragments (not depicted). The parser 304 may parse / entropy-decode the coded video sequence received. The coding of the coded video sequence can be in accordance with a video coding technology or standard, and can follow principles well known to a person skilled in the art, including variable length coding, Huffman coding, arithmetic coding with or without context sensitivity, and so forth. The parser 304 may extract from the coded video sequence, a set of subgroup parameters for at least one of the subgroups of pixels in the video decoder, based upon at least one parameters corresponding to the group. Subgroups can include Groups of Pictures (GOPs), pictures, tiles, slices, macroblocks, Coding Units (CUs), blocks, Transform Units (TUs), Prediction Units (PUs) and so forth. The entropy decoder / parser may also extract from the coded video sequence information such as transform coefficients, quantizer parameter values, motion vectors, and so forth.

[32] The parser 304 may perform entropy decoding / parsing operation on the video sequence received from the buffer 303, so to create symbols 313. The parser 304 may receive encoded data, and selectively decode particular symbols 313. Further, the parser 304 may determine whether the particular symbols 313 are to be provided to a Motion Compensation Prediction unit 306, a scaler / inverse transform unit 305, an Intra Prediction Unit 307, or a loop filter 311.

[33] Reconstruction of the symbols 313 can involve multiple different units depending on the type of the coded video picture or parts thereof (such as: inter and intra picture, inter and intra block), and other factors. Which units are involved, and how, can be controlled by the subgroup control information that was parsed from the coded video sequence by the parser 304. The flow

of such subgroup control information between the parser 304 and the multiple units below is not depicted for clarity.

[34] Beyond the functional blocks already mentioned, decoder 300 can be conceptually subdivided into a number of functional units as described below. In a practical implementation operating under commercial constraints, many of these units interact closely with each other and can, at least partly, be integrated into each other. However, for the purpose of describing the disclosed subject matter, the conceptual subdivision into the functional units below is appropriate.

[35] A first unit is the scaler / inverse transform unit 305. The scaler / inverse transform unit 305 receives quantized transform coefficient as well as control information, including which transform to use, block size, quantization factor, quantization scaling matrices, etc. as symbol(s) 313 from the parser 304. It can output blocks comprising sample values, that can be input into aggregator 310.

[36] In some cases, the output samples of the scaler / inverse transform 305 can pertain to an intra coded block; that is: a block that is not using predictive information from previously reconstructed pictures, but can use predictive information from previously reconstructed parts of the current picture. Such predictive information can be provided by an intra picture prediction unit 307. In some cases, the intra picture prediction unit 307 generates a block of the same size and shape of the block under reconstruction, using surrounding already reconstructed information fetched from the current (partly reconstructed) picture 309. The aggregator 310, in some cases, adds, on a per sample basis, the prediction information the intra prediction unit 307 has generated to the output sample information as provided by the scaler / inverse transform unit 305.

[37] In other cases, the output samples of the scaler / inverse transform unit 305 can pertain to an inter coded, and potentially motion compensated block. In such a case, a Motion Compensation Prediction unit 306 can access reference picture memory 308 to fetch samples used for prediction. After motion compensating the fetched samples in accordance with the symbols 313 pertaining to the block, these samples can be added by the aggregator 310 to the output of the scaler / inverse transform unit (in this case called the residual samples or residual signal) so to generate output sample information. The addresses within the reference picture memory from where the motion compensation unit fetches prediction samples can be controlled by motion vectors, available to the motion compensation unit in the form of symbols 313 that can have, for example X, Y, and reference picture components. Motion compensation also can include interpolation of sample values as fetched from the reference picture memory when sub-sample exact motion vectors are in use, motion vector prediction mechanisms, and so forth.

[38] The output samples of the aggregator 310 can be subject to various loop filtering techniques in the loop filter unit 311. Video compression technologies can include in-loop filter technologies that are controlled by parameters included in the coded video bitstream and made available to the loop filter unit 311 as symbols 313 from the parser 304, but can also be responsive to meta-information obtained during the decoding of previous (in decoding order) parts of the coded picture or coded video sequence, as well as responsive to previously reconstructed and loop-filtered sample values.

[39] The output of the loop filter unit 311 can be a sample stream that can be output to the render device 312 as well as stored in the reference picture memory 557 for use in future inter-picture prediction.

[40] Certain coded pictures, once fully reconstructed, can be used as reference pictures for future prediction. Once a coded picture is fully reconstructed and the coded picture has been identified as a reference picture (by, for example, parser 304), the current reference picture 309 can become part of the reference picture buffer 308, and a fresh current picture memory can be reallocated before commencing the reconstruction of the following coded picture.

[41] The video decoder 300 may perform decoding operations according to a predetermined video compression technology that may be documented in a standard, such as ITU-T Rec. H.265. The coded video sequence may conform to a syntax specified by the video compression technology or standard being used, in the sense that it adheres to the syntax of the video compression technology or standard, as specified in the video compression technology document or standard and specifically in the profiles document therein. Also necessary for compliance can be that the complexity of the coded video sequence is within bounds as defined by the level of the video compression technology or standard. In some cases, levels restrict the maximum picture size, maximum frame rate, maximum reconstruction sample rate (measured in, for example megasamples per second), maximum reference picture size, and so on. Limits set by levels can, in some cases, be further restricted through Hypothetical Reference Decoder (HRD) specifications and metadata for HRD buffer management signaled in the coded video sequence.

[42] In an embodiment, the receiver 302 may receive additional (redundant) data with the encoded video. The additional data may be included as part of the coded video sequence(s). The additional data may be used by the video decoder 300 to properly decode the data and/or to more accurately reconstruct the original video data. Additional data can be in the form of, for example, temporal, spatial, or signal-to-noise ratio (SNR) enhancement layers, redundant slices, redundant pictures, forward error correction codes, and so on.

[43] Fig. 4 may be a functional block diagram of a video encoder 400 according to an embodiment of the present disclosure.

[44] The encoder 400 may receive video samples from a video source 401 (that is not part of the encoder) that may capture video image(s) to be coded by the encoder 400.

[45] The video source 401 may provide the source video sequence to be coded by the encoder (303) in the form of a digital video sample stream that can be of any suitable bit depth (for example: 8 bit, 10 bit, 12 bit, ...), any colorspace (for example, BT.601 Y CrCb, RGB, ...) and any suitable sampling structure (for example Y CrCb 4:2:0, Y CrCb 4:4:4). In a media serving system, the video source 401 may be a storage device storing previously prepared video. In a videoconferencing system, the video source 401 may be a camera that captures local image information as a video sequence. Video data may be provided as a plurality of individual pictures that impart motion when viewed in sequence. The pictures themselves may be organized as a spatial array of pixels, wherein each pixel can comprise one or more samples depending on the sampling structure, color space, etc. in use. A person skilled in the art can readily understand the relationship between pixels and samples. The description below focuses on samples.

[46] According to an embodiment, the encoder 400 may code and compress the pictures of the source video sequence into a coded video sequence 410 in real time or under any other time constraints as required by the application. Enforcing appropriate coding speed is one function of Controller 402. Controller controls other functional units as described below and is functionally coupled to these units. The coupling is not depicted for clarity. Parameters set by controller can include rate control related parameters (picture skip, quantizer, lambda value of rate-distortion optimization techniques, ...), picture size, group of pictures (GOP) layout, maximum motion

vector search range, and so forth. A person skilled in the art can readily identify other functions of controller 402 as they may pertain to video encoder 400 optimized for a certain system design.

[47] Some video encoders operate in what a person skilled in the art readily recognizes as a “coding loop.” As an oversimplified description, a coding loop can consist of the encoding part of an encoder 402 (“source coder” henceforth) (responsible for creating symbols based on an input picture to be coded, and a reference picture(s)), and a (local) decoder 406 embedded in the encoder 400 that reconstructs the symbols to create the sample data that a (remote) decoder also would create (as any compression between symbols and coded video bitstream is lossless in the video compression technologies considered in the disclosed subject matter). That reconstructed sample stream is input to the reference picture memory 405. As the decoding of a symbol stream leads to bit-exact results independent of decoder location (local or remote), the reference picture buffer content is also bit exact between local encoder and remote encoder. In other words, the prediction part of an encoder “sees” as reference picture samples exactly the same sample values as a decoder would “see” when using prediction during decoding. This fundamental principle of reference picture synchronicity (and resulting drift, if synchronicity cannot be maintained, for example because of channel errors) is well known to a person skilled in the art.

[48] The operation of the “local” decoder 406 can be the same as of a “remote” decoder 300, which has already been described in detail above in conjunction with Fig. 3. Briefly referring also to Fig. 4, however, as symbols are available and en/decoding of symbols to a coded video sequence by entropy coder 408 and parser 304 can be lossless, the entropy decoding parts of decoder 300, including channel 301, receiver 302, buffer 303, and parser 304 may not be fully implemented in local decoder 406.

[49] An observation that can be made at this point is that any decoder technology except the parsing/entropy decoding that is present in a decoder also necessarily needs to be present, in substantially identical functional form, in a corresponding encoder. The description of encoder technologies can be abbreviated as they are the inverse of the comprehensively described decoder technologies. Only in certain areas a more detail description is required and provided below.

[50] As part of its operation, the source coder 403 may perform motion compensated predictive coding, which codes an input frame predictively with reference to one or more previously-coded frames from the video sequence that were designated as “reference frames.” In this manner, the coding engine 407 codes differences between pixel blocks of an input frame and pixel blocks of reference frame(s) that may be selected as prediction reference(s) to the input frame.

[51] The local video decoder 406 may decode coded video data of frames that may be designated as reference frames, based on symbols created by the source coder 403. Operations of the coding engine 407 may advantageously be lossy processes. When the coded video data may be decoded at a video decoder (not shown in Fig. 4), the reconstructed video sequence typically may be a replica of the source video sequence with some errors. The local video decoder 406 replicates decoding processes that may be performed by the video decoder on reference frames and may cause reconstructed reference frames to be stored in the reference picture cache 405. In this manner, the encoder 400 may store copies of reconstructed reference frames locally that have common content as the reconstructed reference frames that will be obtained by a far-end video decoder (absent transmission errors).

[52] The predictor 404 may perform prediction searches for the coding engine 407. That is, for a new frame to be coded, the predictor 404 may search the reference picture memory 405 for sample data (as candidate reference pixel blocks) or certain metadata such as reference picture motion vectors, block shapes, and so on, that may serve as an appropriate prediction reference for the new pictures. The predictor 404 may operate on a sample block-by-pixel block basis to find appropriate prediction references. In some cases, as determined by search results obtained by the predictor 404, an input picture may have prediction references drawn from multiple reference pictures stored in the reference picture memory 405.

[53] The controller 402 may manage coding operations of the video coder 403, including, for example, setting of parameters and subgroup parameters used for encoding the video data.

[54] Output of all aforementioned functional units may be subjected to entropy coding in the entropy coder 408. The entropy coder translates the symbols as generated by the various functional units into a coded video sequence, by loss-less compressing the symbols according to technologies known to a person skilled in the art as, for example Huffman coding, variable length coding, arithmetic coding, and so forth.

[55] The transmitter 409 may buffer the coded video sequence(s) as created by the entropy coder 408 to prepare it for transmission via a communication channel 411, which may be a hardware/software link to a storage device which would store the encoded video data. The transmitter 409 may merge coded video data from the video coder 403 with other data to be transmitted, for example, coded audio data and/or ancillary data streams (sources not shown).

[56] The controller 402 may manage operation of the encoder 400. During coding, the controller 405 may assign to each coded picture a certain coded picture type, which may affect

the coding techniques that may be applied to the respective picture. For example, pictures often may be assigned as one of the following frame types:

[57] An Intra Picture (I picture) may be one that may be coded and decoded without using any other frame in the sequence as a source of prediction. Some video codecs allow for different types of Intra pictures, including, for example Independent Decoder Refresh Pictures. A person skilled in the art is aware of those variants of I pictures and their respective applications and features.

[58] A Predictive picture (P picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most one motion vector and reference index to predict the sample values of each block.

[59] A Bi-directionally Predictive Picture (B Picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most two motion vectors and reference indices to predict the sample values of each block. Similarly, multiple-predictive pictures can use more than two reference pictures and associated metadata for the reconstruction of a single block.

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

be coded non-predictively, via spatial prediction or via temporal prediction with reference to one or two previously coded reference pictures.

[61] The video coder 400 may perform coding operations according to a predetermined video coding technology or standard, such as ITU-T Rec. H.265. In its operation, the video coder 400 may perform various compression operations, including predictive coding operations that exploit temporal and spatial redundancies in the input video sequence. The coded video data, therefore, may conform to a syntax specified by the video coding technology or standard being used.

[62] In an embodiment, the transmitter 409 may transmit additional data with the encoded video. The source coder 403 may include such data as part of the coded video sequence. Additional data may comprise temporal/spatial/SNR enhancement layers, other forms of redundant data such as redundant pictures and slices, Supplementary Enhancement Information (SEI) messages, Visual Usability Information (VUI) parameter set fragments, and so on.

[63] Fig. 5 shows a sample DASH client processing model 500, such as of a client sample architecture for processing DASH and Common Media Application Format (CMAF) events, in which a client request of media segments may be based on described addresses in a manifest which also described metadata tracks from which a client may access segments of metadata tracks, parse them, and send them to an application. Further, according to exemplary embodiments, of addresses for media segments such as described below, a DASH manifest may provide addressed for Index segments. Each index segment may provide information about one segment duration and size, and a Representation Index may provide the index information for all segments of a given representation.

[64] Fig. 6 shows an example 600 of a picture-in-picture use case such that a main picture may take up an entire screen, such as a window display or augmented reality view or the like,

while the overlay picture, picture-in-picture, takes a small area of the screen, covering the corresponding area of the main picture. The coordinate of the picture-in-picture (pip) is indicated by x, y, height, and width, where these parameters define the location and size of the pip relative to the main picture coordinate correspondingly.

[65] Viewing the example 700 in Fig. 7 and the flowchart 800 in Fig. 8, it will be understood that according to exemplary embodiments, in the case of streaming, a main video and a pip video may be delivered as two separate streams at S801 such as from different servers or at least different sources such as in video conferencing or the like. And those streams may be delivered as otherwise independent streams to be decoded by separate decoders at S804 after determining No as to the determination at S802, or directly from S801, and then composed together for rendering. But, according to exemplary embodiments, when the used video codec supports merging the streams, it may be determined at S802 such as by a flag included with one or more of the video streams, and/or as metadata thereof, having the respective subpictures at S801, the pip video stream may be combined with the main video stream at S803, possibly replacing the streaming that represents the covered area of the main video with the pip video, and then the single stream is, at S804, sent to the decoder for decoding and then rendering. And thereby a technical burden on the decoder is reduced as the single, merged stream may be delivered for decoding rather than the initial, independent streams to be separately decoded and then merged.

[66] According to exemplary embodiments, such as with example 700 and example 800, VVC subpictures can be used for picture-in-picture services by using both the extraction and merging properties of VVC subpictures. For such a service, the main video is encoded using several subpictures, one of them of the same size as a supplementary video, located at the exact location

where the supplementary video is intended to be composited into the main video and coded independently to enable extraction. If a user chooses, such as at S802, to view the version of the service that includes the supplementary video, the subpicture that corresponds to the picture-in-picture area of the main video is extracted from the main video bitstream, and the supplementary video bitstream is merged with the main video bitstream in its place, as illustrated in the example 700 and example 800 but with the above-described decoding replaced with encoding as will be understood for the purpose of such exemplary embodiments regarding encoding.

[67] And the above-described annotation to one or more subpicture properties of VVC in DASH, such as at S802 and S803, may uses an MPD ContentComponent element to describe the properties of various subpictures of a VVC stream according to exemplary embodiments. For example, a use of such element may be understood from the following Table 1:

Table 1 – Semantics of ContentComponent element used for VVC subpicture annotation according to exemplary embodiments

Element or Attribute Name	Use	Description
ContentComponent		Description of a content component.
@id	0	The id of the subpicture.
@contentType	0	video
@tag	0	The value indicates the property and priority of the subpicture. For picture in picture replacement, the value consists of the string 'PIP' followed by a positive integer number. The number indicates the priority of the corresponding subpicture to be replaced. For example, PIP1 should be the first subpicture to be replaced before PIP2.
Role	0 ... N	The value indicates the application suitability of the subpicture. For instance, the value "sign" for a subpicture shows it is suitable to be replaced with a signed language video.

Element or Attribute Name	Use	Description
Key For attributes: M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory, F=fixed For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are bold ; attributes are non-bold and preceded with an @; list of elements and attributes is in <i>italics bold</i> referring to those taken from the Base type that has been extended by this type.		

[68] And according to exemplary embodiments, any VVC subpicture, a ContentComponent element can be added into the adaptation set or representation, annotating that subpicture. And the DASH client can provide the annotation to a bitstream manipulator to replace the desired subpicture stream with the picture-in-picture video stream and then feed the manipulated VVC bitstream to the VVC decoder according to exemplary embodiments.

[69] Further, embodiments herein extend to other codecs such that the embodiments herein can be used for other video streams that consist of such subpictures, and a same method(s) as described above can be used for any audio or media stream that consists of multiple substreams that can be independently decoded. That is, each substream can be annotated using any one or more of the above methods which thereby at least reduce a coding burden.

[70] The techniques described above, can be implemented as computer software using computer-readable instructions and physically stored in one or more computer-readable media or by a specifically configured one or more hardware processors. For example, FIG. 9 shows a computer system 900 suitable for implementing certain embodiments of the disclosed subject matter.

[71] The computer software can be coded using any suitable machine code or computer language, that may be subject to assembly, compilation, linking, or like mechanisms to create code comprising instructions that can be executed directly, or through interpretation, micro-code

execution, and the like, by computer central processing units (CPUs), Graphics Processing Units (GPUs), and the like.

[72] The instructions can be executed on various types of computers or components thereof, including, for example, personal computers, tablet computers, servers, smartphones, gaming devices, internet of things devices, and the like.

[73] The components shown in FIG. 9 for computer system 900 are exemplary in nature and are not intended to suggest any limitation as to the scope of use or functionality of the computer software implementing embodiments of the present disclosure. Neither should the configuration of components be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary embodiment of a computer system 900.

[74] Computer system 900 may include certain human interface input devices. Such a human interface input device may be responsive to input by one or more human users through, for example, tactile input (such as: keystrokes, swipes, data glove movements), audio input (such as: voice, clapping), visual input (such as: gestures), olfactory input (not depicted). The human interface devices can also be used to capture certain media not necessarily directly related to conscious input by a human, such as audio (such as: speech, music, ambient sound), images (such as: scanned images, photographic images obtain from a still image camera), video (such as two-dimensional video, three-dimensional video including stereoscopic video).

[75] Input human interface devices may include one or more of (only one of each depicted): keyboard 901, mouse 902, trackpad 903, touch screen 910, joystick 905, microphone 906, scanner 908, camera 907.

[76] Computer system 900 may also include certain human interface output devices. Such human interface output devices may be stimulating the senses of one or more human users through, for example, tactile output, sound, light, and smell/taste. Such human interface output devices may include tactile output devices (for example tactile feedback by the touch-screen 910, or joystick 905, but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers 909, headphones (not depicted)), visual output devices (such as screens 910 to include CRT screens, LCD screens, plasma screens, OLED screens, each with or without touch-screen input capability, each with or without tactile feedback capability—some of which may be capable to output two dimensional visual output or more than three dimensional output through means such as stereographic output; virtual-reality glasses (not depicted), holographic displays and smoke tanks (not depicted)), and printers (not depicted).

[77] Computer system 900 can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW 920 with CD/DVD 911 or the like media, thumb-drive 922, removable hard drive or solid state drive 923, legacy magnetic media such as tape and floppy disc (not depicted), specialized ROM/ASIC/PLD based devices such as security dongles (not depicted), and the like.

[78] Those skilled in the art should also understand that term “computer readable media” as used in connection with the presently disclosed subject matter does not encompass transmission media, carrier waves, or other transitory signals.

[79] Computer system 900 can also include interface 999 to one or more communication networks 998. Networks 998 can for example be wireless, wireline, optical. Networks 998 can further be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of networks 998 include local area networks such as Ethernet, wireless LANs,

cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Certain networks 998 commonly require external network interface adapters that attached to certain general-purpose data ports or peripheral buses (950 and 951) (such as, for example USB ports of the computer system 900; others are commonly integrated into the core of the computer system 900 by attachment to a system bus as described below (for example Ethernet interface into a PC computer system or cellular network interface into a smartphone computer system). Using any of these networks 998, computer system 900 can communicate with other entities. Such communication can be uni-directional, receive only (for example, broadcast TV), uni-directional send-only (for example CANbusto certain CANbus devices), or bi-directional, for example to other computer systems using local or wide area digital networks. Certain protocols and protocol stacks can be used on each of those networks and network interfaces as described above.

[80] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core 940 of the computer system 900.

[81] The core 940 can include one or more Central Processing Units (CPU) 941, Graphics Processing Units (GPU) 942, a graphics adapter 917, specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) 943, hardware accelerators for certain tasks 944, and so forth. These devices, along with Read-only memory (ROM) 945, Random-access memory 946, internal mass storage such as internal non-user accessible hard drives, SSDs, and the like 947, may be connected through a system bus 948. In some computer systems, the system bus 948 can be accessible in the form of one or more physical plugs to enable extensions by additional CPUs, GPU, and the like. The peripheral devices can be attached either

directly to the core's system bus 948, or through a peripheral bus 951. Architectures for a peripheral bus include PCI, USB, and the like.

[82] CPUs 941, GPUs 942, FPGAs 943, and accelerators 944 can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM 945 or RAM 946. Transitional data can be also be stored in RAM 946, whereas permanent data can be stored for example, in the internal mass storage 947. Fast storage and retrieval to any of the memory devices can be enabled through the use of cache memory, that can be closely associated with one or more CPU 941, GPU 942, mass storage 947, ROM 945, RAM 946, and the like.

[83] The computer readable media can have computer code thereon for performing various computer-implemented operations. The media and computer code can be those specially designed and constructed for the purposes of the present disclosure, or they can be of the kind well known and available to those having skill in the computer software arts.

[84] As an example and not by way of limitation, an architecture corresponding to computer system 900, and specifically the core 940 can provide functionality as a result of processor(s) (including CPUs, GPUs, FPGA, accelerators, and the like) executing software embodied in one or more tangible, computer-readable media. Such computer-readable media can be media associated with user-accessible mass storage as introduced above, as well as certain storage of the core 940 that are of non-transitory nature, such as core-internal mass storage 947 or ROM 945. The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core 940. A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core 940 and specifically the processors therein (including CPU, GPU, FPGA, and the like) to execute

particular processes or particular parts of particular processes described herein, including defining data structures stored in RAM 946 and modifying such data structures according to the processes defined by the software. In addition or as an alternative, the computer system can provide functionality as a result of logic hardwired or otherwise embodied in a circuit (for example: accelerator 944), which can operate in place of or together with software to execute particular processes or particular parts of particular processes described herein. Reference to software can encompass logic, and vice versa, where appropriate. Reference to a computer-readable media can encompass a circuit (such as an integrated circuit (IC)) storing software for execution, a circuit embodying logic for execution, or both, where appropriate. The present disclosure encompasses any suitable combination of hardware and software.

[85] While this disclosure has described several exemplary embodiments, there are alterations, permutations, and various substitute equivalents, which fall within the scope of the disclosure. It will thus be appreciated that those skilled in the art will be able to devise numerous systems and methods which, although not explicitly shown or described herein, embody the principles of the disclosure and are thus within the spirit and scope thereof.

WHAT IS CLAIMED IS:

1. A method of video coding, the method comprising:
obtaining a dynamic adaptive streaming over HTTP (DASH) video bitstream comprising one or more subpictures;
determining whether the one or more subpictures include Versatile Video Coding (VVC) compliant subpictures;
in response to the DASH video bitstream including the VVC compliant subpictures, annotating the one or more subpictures based on one or more flags; and
manipulating the DASH video stream based on the annotated one or more subpictures.
2. The method according to claim 1,
wherein the one or more subpictures are of a first and second stream of subpictures,
wherein the method comprises controlling a client to replace at least part of one of the first stream and the second stream with another one of the first stream and the second stream based on the annotated one or more subpictures,
wherein the video data comprises dynamic adaptive streaming over HTTP (DASH) video data, and
wherein the client is a DASH client.
3. The method according to claim 2,
wherein the method further comprises determining whether to annotate the one or more of the first subpictures and the second subpictures comprises determining whether to annotate the

one or more of the first subpictures and the second subpictures in a DASH media picture description (MPD).

4. The method according to claim 2,
wherein the other of the first stream and the second stream comprises a picture-in-picture (PIP) stream.

5. The method according to claim 4,
wherein the controlling of the client to replace the at least part of the one of the first stream and the second stream comprises replacing an area represented by a plurality of ones of the first subpictures and the second subpictures with the PIP stream.

6. The method according to claim 5,
wherein the one of the first stream and the second stream represents a main video and the other of the first stream and the second stream represents a separately coded stream as a supplementary video to the main video.

7. The method according to claim 6,
wherein the controlling of the client to replace the at least part of the one of the first stream and the second stream comprises determining whether a user requested to view the supplementary video.

8. The method according to claim 7,

wherein the main video and the supplementary video each comprise versatile video coding (VVC) subpictures.

9. The method according to claim 2,

wherein the controlling of the client to replace the at least part of the one of the first stream and the second stream comprises merging properties of VVC subpictures such that a decoder is provided with a single merged stream merged from both of the first stream and the second stream.

10. The method according to claim 2,

wherein the obtaining of the video data comprises obtaining the first stream separately from the second stream.

11. A apparatus for video streaming, the apparatus comprising:

at least one memory configured to store computer program code;

at least one processor configured to access the computer program code and operate as instructed by the computer program code, the computer program code including:

obtaining code configured to cause the at least one processor to obtain a dynamic adaptive streaming over HTTP (DASH) video bitstream comprising one or more subpictures;

determining code configured to cause the at least one processor to determine whether the one or more subpictures include Versatile Video Coding (VVC) compliant subpictures;

annotating code configured to cause the at least one processor to, in response to the DASH video bitstream including the VVC compliant subpictures, annotate the one or more subpictures based on one or more flags; and

manipulating code configured to cause the at least one processor to manipulate the DASH video stream based on the annotated one or more subpictures.

12. The apparatus according to claim 11,
wherein the one or more subpictures are of a first and second stream of subpictures,
wherein the computer code further includes controlling code configured to cause the at least one processor to control a client to replace at least part of one of the first stream and the second stream with another one of the first stream and the second stream based on the annotated one or more subpictures,
wherein the video data comprises dynamic adaptive streaming over HTTP (DASH) video data, and
wherein the client is a DASH client.

13. The apparatus according to claim 12,
wherein the determining code is further configured to cause the at least one processor to determine whether to annotate the one or more of the first subpictures and the second subpictures in a DASH media picture description (MPD).

14. The apparatus according to claim 12,

wherein the other of the first stream and the second stream comprises a picture-in-picture (PIP) stream.

15. The apparatus according to claim 14,
wherein the controlling code is further configured to cause the at least one processor to replace an area represented by a plurality of ones of the first subpictures and the second subpictures with the PIP stream.

16. The apparatus according to claim 15,
wherein the one of the first stream and the second stream represents a main video and the other of the first stream and the second stream represents a separately coded stream as a supplementary video to the main video.

17. The apparatus according to claim 16,
wherein the controlling code is further configured to cause the at least one processor to determine whether a user requested to view the supplementary video.

18. The apparatus according to claim 17,
wherein the main video and the supplementary video each comprise versatile video coding (VVC) subpictures.

19. The apparatus according to claim 12,

wherein the controlling code is further configured to cause the at least one processor to merge properties of VVC subpictures such that a decoder is provided with a single merged stream merged from both of the first stream and the second stream.

20. A non-transitory computer readable medium storing a program causing a computer to execute a process, the process comprising:

obtaining a dynamic adaptive streaming over HTTP (DASH) video bitstream comprising one or more subpictures;

determining whether the one or more subpictures include Versatile Video Coding (VVC) compliant subpictures;

in response to the DASH video bitstream including the VVC compliant subpictures, annotating the one or more subpictures based on one or more flags; and

manipulating the DASH video stream based on the annotated one or more subpictures.