AN ENCODER, A DECODER AND CORRESPONDING METHODS

~~CROSS-REFERENCE TO RELATED APPLICATIONS~~

~~This application is a continuation of International Application No. PCT/CN2020/070155, filed on January 02, 2020, which is a continuation of International Application No. PCT/CN2019/130581, filed on December 31, 2019. The disclosures of the aforementioned applications are hereby incorporated by reference in their entireties.~~

TECHNICAL FIELD

Embodiments of the present application (disclosure) generally relate to the field of picture processing and more particularly to high-level syntax elements.

BACKGROUND

Video coding (video encoding and decoding) is used in a wide range of digital video applications, for example, broadcast digital TV, video transmission over the internet and mobile networks, real-time conversational applications such as video chat, video conferencing, DVD and Blu-ray discs, video content acquisition and editing systems, and camcorders of security applications.

The amount of video data needed to depict even a relatively short video can be substantial, which may result in difficulties when the data is to be streamed or otherwise communicated across a communications network with limited bandwidth capacity. Thus, video data is generally compressed before being communicated across modern day telecommunications networks. The size of a video could also be an issue when the video is stored on a storage device because memory resources may be limited. Video compression devices often use software and/or hardware at the source to code the video data prior to transmission or storage,

determine the partitioning and prediction mode based on rate distortion optimization (RDO), i.e. select the prediction mode which provides a minimum rate distortion. Terms like “best”, “minimum”, “optimum” etc. in this context do not necessarily refer to an overall “best”, “minimum”, “optimum”, etc. but may also refer to the fulfillment of a termination or selection criterion like a value exceeding or falling below a threshold or other constraints leading potentially to a “sub-optimum selection” but reducing complexity and processing time.

In other words, the partitioning unit 262 may be configured to partition a picture from a video sequence into a sequence of coding tree units (CTUs), and the CTU 203 may be further partitioned into smaller block partitions or sub-blocks (which form again blocks), e.g. iteratively using quad-tree-partitioning (QT), binary-tree--partitioning (BT) or triple-tree-partitioning (TT) or any combination thereof, and to perform, e.g., the prediction for each of the block partitions or sub-blocks, wherein the mode selection comprises the selection of the tree-structure of the partitioned block 203 and the prediction modes are applied to each of the block partitions or sub-blocks.

In the following the partitioning (e.g. by partitioning unit 260) and prediction processing (by inter-prediction unit 244 and intra-prediction unit 254) performed by an example video encoder 20 will be explained in more detail.

Partitioning

The partitioning unit 262 may be configured to partition a picture from a video sequence into a sequence of coding tree units (CTUs), and the partitioning unit 262 may partition (or split) a coding tree unit (CTU) 203 into smaller partitions, e.g. smaller blocks of square or rectangular size. For a picture that has three sample arrays, a CTU consists of an N×N block

have either a square or rectangular shape. For example, the coding tree unit (CTU) is first partitioned by a quaternary tree. Then the quaternary tree leaf nodes can be further partitioned by a multi-type tree structure. There are four splitting types in multi-type tree structure, vertical binary splitting (SPLIT_BT_VER), horizontal binary splitting (SPLIT_BT_HOR), vertical ternary splitting (SPLIT_TT_VER), and horizontal ternary splitting (SPLIT_TT_HOR). The multi-type tree leaf nodes are called coding units (CUs), and unless the CU is too large for the maximum transform length, this segmentation is used for prediction and transform processing without any further partitioning. This means that, in most cases, the CU, PU and TU have the same block size in the quadtree with nested multi-type tree coding block structure. The exception occurs when maximum supported transform length is smaller than the width or height of the colour component of the CU.VVC develops a unique signaling mechanism of the partition splitting information in quadtree with nested multi-type tree coding tree structure. In the signalling smechanism, a coding tree unit (CTU) is treated as the root of a quaternary tree and is first partitioned by a quaternary tree structure. Each quaternary tree leaf node (when sufficiently large to allow it) is then further partitioned by a multi-type tree structure. In the multi-type tree structure, a first flag (mtt_split_cu_flag) is signalled to indicate whether the node is further partitioned; when a node is further partitioned, a second flag (mtt_split_cu_vertical_flag) is signalled to indicate the splitting direction, and then a third flag (mtt_split_cu_binary_flag) is signalled to indicate whether the split is a binary split or a ternary split. Based on the values of mtt_split_cu_vertical_flag and mtt_split_cu_binary_flag, the multi-type tree slitting mode (MttSplitMode) of a CU can be derived by a decoder based on a predefined rule or a table. It should be noted, for a certain design, for example, 64×64 Luma block and 32×32 Chroma block pipelining design in VVC hardware decoders, TT split is forbidden when either width or height of a luma coding block is larger than 64, as shown in Figure 6. TT split is also forbidden when either width or height of a chroma coding block is larger than 32. The pipelining design will divide a picture into

implementation, an encoder 20 can have the quantization unit 208 and the inverse quantization unit 210 combined into a single unit.

Decoder and Decoding Method

Fig. 3 shows an example of a video decoder 30 that is configured to implement the techniques of this present application**.** The video decoder 30 is configured to receive encoded picture data 21 (e.g. encoded bitstream 21), e.g. encoded by encoder 20, to obtain a decoded picture 331. The encoded picture data or bitstream comprises information for decoding the encoded picture data, e.g. data that represents picture blocks of an encoded video slice (and/or tile groups or tiles) and associated syntax elements.

In the example of Fig. 3, the decoder 30 comprises an entropy decoding unit 304, an inverse quantization unit 310, an inverse transform processing unit 312, a reconstruction unit 314 (e.g. a summer 314), a loop filter 320, a decoded picture buffer (DBP) 330, a mode application unit 360, an inter prediction unit 344 and an intra prediction unit 354. Inter prediction unit 344 may be or include a motion compensation unit. Video decoder 30 may, in some examples, perform a decoding pass generally reciprocal to the encoding pass described with respect to video encoder 100 from FIG. 2.

As explained with regard to the encoder 20, the inverse quantization unit 210, the inverse transform processing unit 212, the reconstruction unit 214, the loop filter 220, the decoded picture buffer (DPB) 230, the inter prediction unit 344 and the intra prediction unit 354 are also referred to as forming the “built-in decoder” of video encoder 20. Accordingly, the inverse quantization unit 310 may be identical in function to the inverse quantization unit ~~110~~210, the inverse transform processing unit 312 may be identical in function to the inverse transform processing unit 212, the reconstruction unit 314 may be identical in function to

Similarly, it is provided a method of encoding a video bitstream comprising coded data implemented in an encoder as illustrated in Figure 9. The method comprises: S901, determining whether an adaptive loop filter (ALF) is enabled for the current picture. S902~~3~~, encoding a picture-level syntax element (e.g. pic_alf_enabled_flag) into a picture header of the current picture based on the determining whether an adaptive loop filter(ALF) is enabled for the current picture, wherein the picture-level syntax element is used to specify whether an adaptive loop filter(ALF) is enabled for the current picture, and S903~~5~~, in case that an adaptive loop filter is enabled for the current picture, encoding a parameter set identifier (e.g. pic_alf_aps_id_luma[i]) into the picture header, wherein the parameter set identifier is used to indicate a first parameter set (e.g. an APS) comprising a set of ALF parameters of an adaptive loop filter, and wherein a temporal identifier (e.g. TemporalId) of the first parameter set is set to be less than or equal to a temporal identifier of the current picture.

The method may further comprise: encoding the temporal identifier of the first parameter set and the temporal identifier of the current picture into the video bitstream.

The first parameter set comprises a set of ALF parameters of an adaptive loop filter performed on the luma component of at least one slice of the current picture.

The method may further comprise: in case that the picture-level syntax element specifies the adaptive loop filter is enabled for the current picture, encoding another parameter set identifier into the picture header, wherein the parameter set identifier (e.g. pic_alf_aps_id_chroma) is used to indicate a second parameter set (e.g. an APS) comprising a set of ALF parameters of an adaptive loop filter performed on the chroma component of at

CLAIMS

1. A method for decoding of a video bitstream implemented by a decoding device, wherein the video bitstream includes data representing a current picture, the method comprising:

obtaining a picture-level syntax element included in a picture header of the current picture by parsing the video bitstream, wherein the picture-level syntax element is used to specify whether an adaptive loop filter(ALF) is enabled for the current picture;

in case that the picture-level syntax element specifies the adaptive loop filter is enabled for the current picture, obtaining a first parameter set identifier included in the picture header, wherein the first parameter set identifier is used to indicate a first parameter set comprising a set of ALF parameters, and wherein a temporal identifier of the first parameter set is less than or equal to a temporal identifier of the current picture; and

performing the adaptive loop filter on at least one slice of the current picture based on the first parameter set.

2. The method of claim 1, wherein the performing the adaptive loop filter on at least one slice of the current picture based on the first parameter set comprises: performing adaptive loop filter on the luma component of at least one slice of the current picture based on the first parameter set.

3. The method of claim 2, wherein the method further comprises: in case that the picture-level syntax element specifies the adaptive loop filter is enabled for the current picture, obtaining ~~another~~a second parameter set identifier included in the picture header, wherein the second parameter set identifier is used to indicate a second parameter set comprising a set of ALF parameters, and wherein a temporal identifier of the second parameter set is less than or equal to ~~a~~ the temporal identifier of the current picture; and performing adaptive loop filter on

the chroma component of at least one slice of the current picture based on the second parameter set.

4. A method for encoding of a video bitstream implemented by an encoding device, wherein the video bitstream includes data representing a current picture, the method comprising:

determining whether an adaptive loop filter(ALF) is enabled for the current picture;

encoding a picture-level syntax element into a picture header of the current picture based on the determining whether an adaptive loop filter(ALF) is enabled for the current picture, wherein the picture-level syntax element is used to specify whether an adaptive loop filter(ALF) is enabled for the current picture; and

in case that an adaptive loop filter is enabled for the current picture, encoding a first parameter set identifier into the picture header, wherein the first parameter set identifier is used to indicate a first parameter set comprising a set of ALF parameters of an adaptive loop filter, and wherein a temporal identifier of the first parameter set is set to be less than or equal to a temporal identifier of the current picture.

5. The method of claim 4, wherein the method further comprises: encoding the temporal identifier of the first parameter set and the temporal identifier of the current picture into the video bitstream.

6. The method of claim 4 or 5, wherein the first parameter set comprises a set of ALF parameters of an adaptive loop filter performed on the luma component of at least one slice of the current picture.

7. The method of claim 6, wherein the method further comprises: in case that the picture-level syntax element specifies the adaptive loop filter is enabled for the current picture, encoding ~~another~~a second parameter set identifier into the picture header, wherein the second parameter set identifier is used to indicate a second parameter set comprising a set of ALF parameters of an adaptive loop filter performed on the chroma component of at least one slice of the current picture, and wherein a temporal identifier of the second parameter set is set to be less than or equal to ~~a~~ the temporal identifier of the current picture.

8. An encoder comprising processing circuitry for carrying out the method according to any one of claims 4 to 7.

9. A decoder comprising processing circuitry for carrying out the method according to any one of claims 1 to 3.

10. A computer program product comprising program code for performing the method according to any one of the preceding claims when executed on a computer or a processor.

11. A decoder, comprising:

one or more processors; and

a non-transitory computer-readable storage medium coupled to the processors and storing programming for execution by the processors, wherein the programming, when executed by the processors, configures the decoder to carry out the method according to any one of the ~~preceding~~ claims 1-3.

12. An encoder, comprising:

one or more processors; and

a non-transitory computer-readable storage medium coupled to the processors and ~~—~~storing programming for execution by the processors, wherein the programming, when executed by the processors, configures the encoder to carry out the method according to any one of the ~~preceding~~ claims 4-7.

13. A non-transitory computer-readable medium carrying a program code which, when executed by a computer device, causes the computer device to perform the method of any one of the preceding claims.

14. A non-transitory storage medium which includes an encoded bitstream decoded by an image decoding device, the bitstream being generated by dividing a current picture of a video signal or an image signal into a plurality blocks, and comprising a plurality of syntax elements, wherein the plurality of syntax elements comprises a picture-level syntax element included in a picture header of the current picture by parsing the video bitstream, wherein the picture-level syntax element is used to specify whether an adaptive loop filter(ALF) is enabled for the current picture; in case that the picture-level syntax element specifies the adaptive loop filter is enabled for the current picture, the bitstream further comprises a first ~~n~~ parameter set identifier included in the picture header, wherein the first parameter set identifier is used to indicate a first parameter set comprising a set of ALF parameters, and wherein a temporal identifier of the first parameter set is set to be less than or equal to a temporal identifier of the current picture.

15. A video data decoding device, comprising:

a non-transitory memory storage, configured to store video data in a form of a bitstream; and

a decoder, configured to perform any of the methods according to any one of the claims 1 to 3.

16. A video data encoding device, comprising:

a non-transitory memory storage, configured to store video data in a form of a bitstream; and

an encoder, configured to perform any of the methods according to any one of the claims 4 to 7.