neighboring blocks in the same picture or temporal prediction with respect to reference samples in other reference pictures. Pictures may be referred to as frames, and reference pictures may be referred to a reference frames.

[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] In general, this disclosure describes techniques for context adaptive binary arithmetic coding (CABAC) in a video coding process. In particular, this disclosure proposes a reduction in the number of CABAC contexts used for one or more syntax elements, non-limiting examples of which include *pred_type, merge_idx, inter_pred_flag, ref_idx_lx, cbf_cb, cbf_cr, coeff_abs_level_greater1_flag,* and *coeff_abs_level_greater2_flag*. The modifications may reduce up to 56 contexts with negligible coding efficiency changes. The proposed context reductions for the syntax elements may be used alone or in any combination.

[0007] In one example of the disclosure, a method of encoding video may include determining a first prediction type for a block of video data in a P slice, representing the first prediction type as a P-slice prediction type syntax element, determining a second prediction type for a block of video data in a B slice, representing the second prediction type as a B-slice prediction type syntax element, determining a P-slice binarization for the P-slice prediction type syntax element, determining a B-slice binarization for the B-slice prediction type syntax element, wherein the P-slice prediction type syntax element and the B-slice prediction type syntax element are determined using the same

binarization logic, and encoding the video data based on the binarizations of the P-slice prediction type syntax element and the B-slice prediction syntax element.

[0008] In another example of the disclosure, a method of decoding video may include mapping a binarized P-slice prediction type syntax element to a prediction type using a binarization mapping for a block of video data in a P slice, mapping a binarized B-slice prediction type syntax element to a prediction type using the same binarization mapping for a block of video data in a B slice, and decoding the video data based on the mapped prediction types.

[0009] In another example of the disclosure, a method of encoding video data comprises determining a partition type for a prediction mode for a block of video data, encoding a partition type bin of a prediction type syntax element for a block of video data using CABAC with a single context, wherein the single context is the same for any partition type, and encoding a partition size bin of the prediction type syntax element for the block of video data using CABAC in bypass mode.

[0010] In another example of the disclosure, a method of decoding video data comprises receiving a prediction type syntax element for a block of video data that has been coded using CABAC, the prediction type syntax element including a partition type bin representing a partition type and a partition size bin representing a partition size, decoding the partition type bin of the prediction type syntax element using context adaptive binary arithmetic coding with a single context, wherein the single context is the same for any partition type, and decoding the partition size bin of the prediction type syntax element using CABAC in bypass mode.

[0011] In another example of the disclosure, a method of coding video data comprises coding a Cb chroma coded block flag for a block of video data using CABAC, wherein coding the Cb chroma coded block flag comprises using a context set including one or more contexts as part of the CABAC, and coding a Cr chroma coded block flag using CABAC, wherein coding the Cr chroma coded block flag comprises using the same context set as the Cb chroma coded block flag as part of the CABAC.

[0012] This disclosure also describes the above techniques in terms of apparatuses configured to perform the techniques as well as in terms of a computer-readable storage medium storing instructions that, when executed, cause one or more processors to perform the techniques.

another bin string is used for an intra-predicted block with an NxN partition type. As shown in Table 1, the bin string used for I slices are not dependent on CU size.

[0076] For P and B slices, in Table 1, different bin strings are used for each value of pred_type. Again, the value of pred_type depends on both the prediction mode (inter-prediction or intra-prediction) and the partition type used. For P and B slices, the actual bin string used further depends on the size of the CU being coded and whether or not inter-prediction is enabled for a 4x4 block size.

[0077] The first column under bin string applies for the situation where the logarithmic function of the CU size of the CU being coded is greater than the logarithmic function of the minimum allowable CU size. According to one example in HEVC, the first column of bin strings is used if cLog2CUSize > Log2MinCUsize. The logarithmic function is used to create a smaller number so that a smaller consecutive index may be used.

[0078] If the logarithmic function of the CU size of the CU being coded is equivalent to the logarithmic function the minimum allowable CU size (i.e., cLog2CUSize = = Log2MinCUSize) then one of columns 2 and 3 under bin string in Table 1 is used to select the binarization. Column 2 is used when the logarithmic function of the CU size of the CU being coded is equivalent to 3 and inter-prediction for a 4x4 CU is not enabled (i.e., cLog2CUSize = = 3 && !inter_4x4_enabled_flag). Column 3 is used when the logarithmic function of the CU size of the CU being coded is greater than 3 or when inter-prediction for a 4x4 CU is enabled (i.e., cLog2CUSize > 3 | | inter_4x4_enabled_flag).

[0079] Table 2 below shows example binarizations where P and B slices use the same bin strings, in accordance with one or more examples described in this disclosure. As shown in Table 2, P slices use the same binarizations used for B slices in Table 1. In this way, it is not necessary to store and use a separate set of contexts for both P and B slices. As such, the total number of contexts needed to code the *pred_type* syntax element is reduced. Furthermore, only one mapping (instead of two) between the bin string logic (shown in columns (1) – (3)) and the actual bin string need be stored.

coding loop or after the coding loop) may also be used to smooth pixel transitions, or otherwise improve the video quality. The decoded video blocks in a given frame or picture are then stored in reference picture memory 92, which stores reference pictures used for subsequent motion compensation. Reference picture memory 92 also stores decoded video for later presentation on a display device, such as display device 32 of FIG. 1.

[0147] FIG. 6 is a flowchart illustrating an example video encoding method of the disclosure. The method of FIG. 6 may be implemented by video encoder 20. Video encoder 20 may be configured to determine a first prediction type for a block of video data in a P slice (602), and to represent the first prediction type as a P-slice prediction type syntax element (604). Video encoder 20 may be further configured to determine a second prediction type for a block of video data in a B slice (606), and to represent the second prediction type as a B-slice prediction type syntax element (608). The P-slice prediction type syntax element and the B-slice prediction type syntax element specify a prediction mode and a partition type. The prediction mode may include one of inter-prediction and intra-prediction. The partition type may include one of symmetric partitions and asymmetric partitions.

[0148] Video encoder 20 may be further configured to determine a P-slice binarization for the P-slice prediction type syntax element (610), and to determine a B-slice binarization for the B-slice prediction type syntax element, wherein the P-slice prediction type syntax element and the B-slice prediction type syntax element are determined using the same binarization logic (612). Video encoder 20 may then encode the video data based on the binarizations of the P-slice prediction type syntax element and the B-slice prediction syntax element (614).

[0149] Encoding the video data may comprise binarizing the P-slice prediction type syntax element with the determined P-slice binarizations, binarizing the B-slice prediction type syntax element with the determined B-slice binarizations, applying context adaptive binary arithmetic coding (CABAC) to the binarized P-slice prediction type syntax element, and applying context adaptive binary arithmetic coding (CABAC) to the binarized B-slice prediction type syntax element.

[0150] FIG. 7 is a flowchart illustrating an example video decoding method of the disclosure. The method of FIG. 7 may be implemented by video decoder 30. Video decoder 30 may be configured to receive a context adaptive binary arithmetic coded P-slice prediction type syntax element which indicates the prediction type for the block of

neighboring blocks in the same picture or temporal prediction with respect to reference samples in other reference pictures. Pictures may be referred to as frames, and reference pictures may be referred to as reference frames.

[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] In general, this disclosure describes techniques for context adaptive binary arithmetic coding (CABAC) in a video coding process. In particular, this disclosure proposes a reduction in the number of CABAC contexts used for one or more syntax elements, non-limiting examples of which include *pred_type, merge_idx, inter_pred_flag, ref_idx_lx, cbf_cb, cbf_cr, coeff_abs_level_greater1_flag,* and *coeff_abs_level_greater2_flag*. The modifications may reduce up to 56 contexts with negligible coding efficiency changes. The proposed context reductions for the syntax elements may be used alone or in any combination.

[0007] In one example of the disclosure, a method of encoding video may include determining a first prediction type for a block of video data in a P slice, representing the first prediction type as a P-slice prediction type syntax element, determining a second prediction type for a block of video data in a B slice, representing the second prediction type as a B-slice prediction type syntax element, determining a P-slice binarization for the P-slice prediction type syntax element, determining a B-slice binarization for the B-slice prediction type syntax element, wherein the P-slice prediction type syntax element and the B-slice prediction type syntax element are determined using the same

binarization logic, and encoding the video data based on the binarizations of the P-slice prediction type syntax element and the B-slice prediction type syntax element.

[0008] In another example of the disclosure, a method of decoding video may include mapping a binarized P-slice prediction type syntax element to a prediction type using a binarization mapping for a block of video data in a P slice, mapping a binarized B-slice prediction type syntax element to a prediction type using the same binarization mapping for a block of video data in a B slice, and decoding the video data based on the mapped prediction types.

[0009] In another example of the disclosure, a method of encoding video data comprises determining a partition type for a prediction mode for a block of video data, encoding a partition type bin of a prediction type syntax element for a block of video data using CABAC with a single context, wherein the single context is the same for any partition type, and encoding a partition size bin of the prediction type syntax element for the block of video data using CABAC in bypass mode.

[0010] In another example of the disclosure, a method of decoding video data comprises receiving a prediction type syntax element for a block of video data that has been coded using CABAC, the prediction type syntax element including a partition type bin representing a partition type and a partition size bin representing a partition size, decoding the partition type bin of the prediction type syntax element using context adaptive binary arithmetic coding with a single context, wherein the single context is the same for any partition type, and decoding the partition size bin of the prediction type syntax element using CABAC in bypass mode.

[0011] In another example of the disclosure, a method of coding video data comprises coding a Cb chroma coded block flag for a block of video data using CABAC, wherein coding the Cb chroma coded block flag comprises using a context set including one or more contexts as part of the CABAC, and coding a Cr chroma coded block flag using CABAC, wherein coding the Cr chroma coded block flag comprises using the same context set as the Cb chroma coded block flag as part of the CABAC.

[0012] This disclosure also describes the above techniques in terms of apparatuses configured to perform the techniques as well as in terms of a computer-readable storage medium storing instructions that, when executed, cause one or more processors to perform the techniques.

another bin string is used for an intra-predicted block with an NxN partition type. As shown in Table 1, the bin string used for I slices are not dependent on CU size.

[0076] For P and B slices, in Table 1, different bin strings are used for each value of pred_type. Again, the value of pred_type depends on both the prediction mode (inter-prediction or intra-prediction) and the partition type used. For P and B slices, the actual bin string used further depends on the size of the CU being coded and whether or not inter-prediction is enabled for a 4x4 block size.

[0077] The first column under bin string applies for the situation where the logarithmic function of the CU size of the CU being coded is greater than the logarithmic function of the minimum allowable CU size. According to one example in HEVC, the first column of bin strings is used if cLog2CUSize > Log2MinCUsize. The logarithmic function is used to create a smaller number so that a smaller consecutive index may be used.

[0078] If the logarithmic function of the CU size of the CU being coded is equivalent to the logarithmic function of the minimum allowable CU size (i.e., cLog2CUSize = = Log2MinCUSize) then one of columns 2 and 3 under bin string in Table 1 is used to select the binarization. Column 2 is used when the logarithmic function of the CU size of the CU being coded is equivalent to 3 and inter-prediction for a 4x4 CU is not enabled (i.e., cLog2CUSize = = 3 && !inter_4x4_enabled_flag). Column 3 is used when the logarithmic function of the CU size of the CU being coded is greater than 3 or when inter-prediction for a 4x4 CU is enabled (i.e., cLog2CUSize > 3 | | inter_4x4_enabled_flag).

[0079] Table 2 below shows example binarizations where P and B slices use the same bin strings, in accordance with one or more examples described in this disclosure. As shown in Table 2, P slices use the same binarizations used for B slices in Table 1. In this way, it is not necessary to store and use a separate set of contexts for both P and B slices. As such, the total number of contexts needed to code the *pred_type* syntax element is reduced. Furthermore, only one mapping (instead of two) between the bin string logic (shown in columns (1) – (3)) and the actual bin string need be stored.

coding loop or after the coding loop) may also be used to smooth pixel transitions, or otherwise improve the video quality. The decoded video blocks in a given frame or picture are then stored in reference picture memory 92, which stores reference pictures used for subsequent motion compensation. Reference picture memory 92 also stores decoded video for later presentation on a display device, such as display device 32 of FIG. 1.

[0147] FIG. 6 is a flowchart illustrating an example video encoding method of the disclosure. The method of FIG. 6 may be implemented by video encoder 20. Video encoder 20 may be configured to determine a first prediction type for a block of video data in a P slice (602), and to represent the first prediction type as a P-slice prediction type syntax element (604). Video encoder 20 may be further configured to determine a second prediction type for a block of video data in a B slice (606), and to represent the second prediction type as a B-slice prediction type syntax element (608). The P-slice prediction type syntax element and the B-slice prediction type syntax element specify a prediction mode and a partition type. The prediction mode may include one of inter-prediction and intra-prediction. The partition type may include one of symmetric partitions and asymmetric partitions.

[0148] Video encoder 20 may be further configured to determine a P-slice binarization for the P-slice prediction type syntax element (610), and to determine a B-slice binarization for the B-slice prediction type syntax element, wherein the P-slice prediction type syntax element and the B-slice prediction type syntax element are determined using the same binarization logic (612). Video encoder 20 may then encode the video data based on the binarizations of the P-slice prediction type syntax element and the B-slice prediction type syntax element (614).

[0149] Encoding the video data may comprise binarizing the P-slice prediction type syntax element with the determined P-slice binarizations, binarizing the B-slice prediction type syntax element with the determined B-slice binarizations, applying context adaptive binary arithmetic coding (CABAC) to the binarized P-slice prediction type syntax element, and applying context adaptive binary arithmetic coding (CABAC) to the binarized B-slice prediction type syntax element.

[0150] FIG. 7 is a flowchart illustrating an example video decoding method of the disclosure. The method of FIG. 7 may be implemented by video decoder 30. Video decoder 30 may be configured to receive a context adaptive binary arithmetic coded P-slice prediction type syntax element which indicates the prediction type for the block of