**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

**APPLICATION FOR PROVISIONAL LETTERS PATENT**

**HIGH LEVEL SYNTAX FOR STATIC SCREEN CONTENT CODING IN HIGH EFFICIENCY VIDEO CODING**

BY:

Thorsten Laude

Am kleinen Felde 29, 30167

Hannover, Germany

Citizenship: German

Joern Ostermann

Karla-Schmidt-Strasse 14, 30655

Hannover, Germany

Citizenship: German

AND

Haoping Yu

3471 Glen Abbe Court

Carmel, IN 46032

Citizenship: United States

**High Level Syntax For Static Screen Content Coding In High Efficiency Video Coding**

**DETAILED DESCRIPTION**

1. It should be understood at the outset that although an illustrative implementation of one or more embodiments are provided below, the disclosed systems and/or methods may be implemented using any number of techniques, whether currently known or in existence. The disclosure should in no way be limited to the illustrative implementations, drawings, and techniques illustrated below, including the exemplary designs and implementations illustrated and described herein, but may be modified within the scope of the appended claims along with their full scope of equivalents.
2. With the recent growth of cloud-based services and the substitution of conventional computers by mobile devices such as smartphones and tablet computers new scenarios emerge where computer generated content, or screen content, is generated on one device but displayed using a second device. One possible scenario is that of an application running on a remote server with the display output being displayed on the local workstation of the user. Another scenario is the duplication of a smartphone or tablet computer screen to the screen of a television device, e.g. with the purpose of watching a movie on the big screen rather than on the small screen of the mobile device.
3. These scenarios are accompanied by the need of an efficient transmission of screen content which should be capable of representing the screen content video with sufficient visual quality while observing data rate constraints given by existing transmission systems. A suitable solution for this challenge could be the usage of video coding technologies to compress the screen content. These video coding technologies have been well studied during the last decades, as described in D. Salomon and G. Motta, Handbook of Data Compression, 5th ed. London: Springer Verlag, 2010, which is incorporated herein by this reference. These video coding technologies have resulted in several often used video coding standards like MPEG-2, as described in ISO/IEC 13818–2, Generic coding of moving pictures and associated audio information—Part 2: Video/ITU-T Recommendation H.262, 1994, and B. G. Haskell, A. Puri, and A. N. Netravali, Digital Video: An Introduction to MPEG-2, New York: Chapman & Hall, 1997, and MPEG-4, as described in ISO/IEC 14496: MPEG-4 Coding of audio-visual objects, F. Pereira and T. Ebrahimi, The MPEG-4 book, Upper Saddle River, New Jersey, USA: Prentice Hall PTR, 2002, and A. Puri and T. Chen, Multimedia Systems, Standards, and Networks, New York: Marcel Dekker, Inc., 2000, and Advanced Video Coding (AVC), as described in ISO/IEC 14496–10, Coding of Audiovisual Objects-Part 10: Advanced Video Coding/International Telecommunication Union Telecommunication Standardization Sector (ITU-T) Recommendation H.264 Advanced video coding for generic audiovisual services, 2003, each of which is incorporated herein by this reference.
4. Recently, the Joint Collaborative Team on Video Coding (JCT-VC) of the Moving Pictures Expert Group (MPEG) and of the Video Coding Experts Group (VCEG) developed the successor of AVC which is called High Efficiency Video Coding (HEVC), as described in ITU-T Recommendation H.265/ISO/IEC 23008-2:2013 MPEG-H Part 2: High Efficiency Video Coding (HEVC), 2013, and M. Wien, High Efficiency Video Coding - Coding Tools and Specification, 1st ed. Berlin Heidelberg: Springer, 2015, each of which is incorporated herein by this reference. HEVC is based upon the same concept of hybrid video coding as AVC but achieves a compression performance twice as good as the predecessor standard by improving the existing coding tools and adding new coding tools, as described in P. Hanhart, M. Rerabek, F. De Simone, and T. Ebrahimi, “Subjective quality evaluation of the upcoming HEVC video compression standard,” in SPIE Optical Engineering + Applications, 2012, p. 84990V, which is incorporated herein by this reference.
5. However, HEVC has been developed with the aim of compressing natural, i.e. camera captured, content. The consequence is that HEVC provides superior compression performance for natural content but possibly is not the best solution to compress screen content. Thus, after finalizing Version 1 of HEVC, a Call for Proposals for Screen Content Coding (SCC) has been issued by the JCT-VC in January 2014. Responses to this call provided more sophisticated compression methods specifically designed for screen content, as described in J. Chen, Y. Chen, T. Hsieh, R. Joshi, M. Karczewicz, W.-S. Kim, X. Li, C. Pang, W. Pu, K. Rapaka, J. Sole, L. Zhang, and F. Zou, JCT-VC Q0031: Description of screen content coding technology proposal by Qualcomm. 17th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11. Valencia, ES, 27 March – 4 April. 2014, C.-C. Chen, T.-S. Chang, R.-L. Liao, C.-W. Kuo, W.-H. Peng, H.-M. Hang, Y.-J. Chang, C.-H. Hung, C.-C. Lin, J.-S. Tu, K. Erh-Chung, J.-Y. Kao, C.-L. Lin, and F.-D. Jou, JCT-VC Q0032: Description of screen content coding technology proposal by NCTU and ITRI International, 17th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11. Valencia, ES, 27 March – 4 April 2014, P. Lai, T.-D. Chuang, Y.-C. Sun, X. Xu, J. Ye, S.-T. Hsiang, Y.-W. Chen, K. Zhang, X. Zhang, S. Liu, Y.-W. Huang, and S. Lei, JCT-VC Q0033: Description of screen content coding technology proposal by MediaTek, 17th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Valencia, ES, 27 March – 4 April 2014, Z. Ma, W. Wang, M. Xu, X. Wang, and H. Yu, JCT-VC Q0034: Description of screen content coding technology proposal by Huawei. 17th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Valencia, ES, 27 March – 4 April. 2014, and B. Li, J. Xu, F. Wu, X. Guo, and G. J. Sullivan, JCT-VC Q0035: Description of screen content coding technology proposal by Microsoft, 17th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Valencia, ES, 27 March – 4 April 2014, each of which is incorporated herein by this reference.
6. It is worth noting that natural content and screen content videos may have characteristics that differ significantly in terms of edge sharpness, amount of different colors among other properties as it has been studied in T. Lin, P. Zhang, S. Wang, K. Zhou, and X. Chen, “Mixed Chroma Sampling-Rate High Efficiency Video Coding for Full-Chroma Screen Content,” Institute of Electrical and Electronics Engineers (IEEE) Trans. Circuits Syst. Video Technol., vol. 23, no. 1, pp. 173–185, Jan. 2013, which is incorporated herein by this reference. Another typical characteristic of screen content videos may be the absence of changes between consecutive pictures or parts of these pictures in such videos. One possible scenario among a variety of other scenarios where such unchanged areas may appear is static background in screen content. Due to the typical absence of noise the unchanged areas may be identical in terms of sample values.
7. In this disclosure we present methods which may be used to code static screen content. It is noted that all described methods may be applicable not only for static screen content but for any video signals and that the coding of static screen content is only used as one application example for the described methods.
8. Described below is proposed technology for high level syntax for static screen content coding. There may be a scenario where some part of the current picture may be static compared to the corresponding parts in previously coded pictures. It may further be beneficial to use the corresponding parts in these previously coded pictures to code the parts in the current picture. Therefore, the static part in the current picture may be coded by copying the corresponding part from a previously coded picture or several previously coded pictures. The corresponding part may be the area in the previously coded picture which is at the same position as the part in the current picture.
9. Furthermore, a picture may be partitioned into slices, as described in G. J. Sullivan, J.-R. Ohm, W. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard,” IEEE TRANS. CIRCUITS Syst. VIDEO Technol., Dec. 2012, which is incorporated herein by this reference. FIG. 1 illustrates an example for the partitioning of a picture into slices. In this particular example, the picture is partitioned into five slices. Moreover, a picture may be partitioned into tiles. The partitioning of a picture into slices and tiles may be combined, e.g. by further partitioning a slice into tiles or by further partitioning a tile into slices.



FIG. 1 – Example for the partitioning of a picture into five slices

1. FIG. 2 illustrates an example in which the picture is partitioned into five slices as in FIG. 1. In this example, each of the slices is further partitioned into two tiles. It is worth noting that the partitioning into slices of equal size and the partitioning of these slices into tiles of equal size is arbitrary and no requirement for the described methods. Other ways of partitioning a picture may be applied. Additionally, a slice may be further portioned into slice segments.



FIG. 2 – Example for the partitioning of a picture into slices and tiles

1. The methods which are described in the following may be applied to areas in a picture. Slices, slice segments, tiles or a combination thereof are only used as examples to further illustrate the methods. However, none of the methods is limited to these embodiments of partitions within a picture. For the sake of an easy readability, the term “area” will be used in the following to denote “slice, slice segment or tile”.
2. Additionally, it is evident that an area may be static compared to the corresponding area in a previously coded picture. Moreover, the corresponding area in the previously coded picture may belong to an area with the same size and location as the area in the current picture. This knowledge may be utilized for the efficient coding of the static area. For instance, the area from the previous picture may be copied to the current picture. This way, the area of the current picture does not need to be coded. An example for this method is illustrated in FIG. 3. As in the previous example, a picture is partitioned into five slices. Although the previously coded picture (denoted with time instance t-n) is partitioned into slices in the same manner as the current picture (denoted with time instance t), it is noteworthy that this is no requirement for the presented method. In the example, it is assumed that the first three slices are static. In consequence, these three slices are copied from the previously coded picture to the current picture.



FIG. 3 - Example for the copy of three slices from a previously coded picture to the current picture

1. Several embodiments for the slice or slice segment copy are possible and described in the following. As one embodiment the complete slice or slice segment including the sample values and other syntax elements such as slice segment header, coding unit (CU), prediction unit (PU) and transform unit (TU) syntax may be copied. As another embodiment, only a part of the slice or slice segment may be copied. For instance, the slice segment header may be copied while the remaining syntax elements are signaled for the new picture or the slice segment header may be signaled while the remaining syntax is copied. Other embodiments of partial slice or slice segment copies are possible.
2. Additionally, various embodiments for the tile copy are possible. For instance, the complete tile including all syntax elements and all sample values may be copied. As another embodiment, partial tile copies are possible, e.g. only sample values.
3. Furthermore, it may be specified that copied areas may be interpreted as intra coded during the coding of subsequent pictures. This way, only sample values might be copied for this particular area.
4. There may be a scenario in which the location of an area in the current picture corresponds to the locations of several areas in the previously coded picture. In such a case, it may be beneficial to copy the information for the area in the current picture from multiple areas in the previously coded picture. For instance, the slice segment header may be copied from one slice segment while syntax elements of the CU, PU and TU syntax are copied from multiple areas.
5. Furthermore, different approaches to indicate the area copy may exist. As one embodiment, the area copy may be indicated for the current area. However, this may not be the optimal approach. For instance, in typical screen content applications the static part may be static for many consecutive pictures. Thus, it is undesirable to indicate the copy in each of the consecutive pictures with a static part. Additionally, considering that many screen content applications require low latency, it may not be possible to determine for how long the content of the area may be static. This may be the case because the analysis of subsequent pictures would cause a delay for the coding of the current picture. Thus, a copy process of indeterminate duration may be beneficial. For this reason, a “freeze” operation (e.g., slice freeze, slice segment freeze, tile freeze) is introduced as another embodiment of our method.
6. The freeze operation may be indicated for an area of the current picture. If the freeze operation is indicated for an area, this area may be copied to subsequently coded pictures. As already elaborated above, the copy operation may include the complete area or parts of the area syntax (e.g., only sample values).
7. The usage of the freeze process may be signaled as part of the bitstream, e.g. by a flag as part of the slice segment header, in the slice segment data, in the picture parameter set (PPS), in the sequence parameter set (SPS) or in the video parameter set (VPS). Table 1 shows an example how the freeze process may be signaled as part of the slice segment header syntax. In this particular example, the syntax element slice\_freeze\_flag may indicate the usage of the freeze process. For instance, the freeze process may be used if the syntax element is equal to 1 while the freeze mode may not be used if the syntax element is equal to 0. It may further be specified that the value of slice\_freeze\_flag may be inferred as 0 if the syntax element is not present in the bitstream. Other syntax may be applied to signal the freeze process. Changes relative to the HEVC screen content coding draft, as described in R. Joshi and J. Xu, JCT-VC S1005: High Efficiency Video Coding (HEVC) Screen Content Coding: Draft 2, 19th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC), Strasbourg, FR, 2014, which is incorporated herein by this reference, are highlighted.

|  |  |
| --- | --- |
| slice\_segment\_header( ) { | **Descriptor** |
| […] |  |
| **slice\_freeze \_flag** | u(1) |
| byte\_alignment( ) |  |
| } |  |

Table 1 – Example for the freeze process syntax as part of the slice segment header

1. At the time instance, when the freeze process is signaled as being used for an area, it is uncertain for how many subsequent pictures the area will be copied. Thus, there are various possibilities how to determine for how many pictures the area should be copied whereof some examples are described in the following. As one embodiment of our method, the area may be copied to subsequent pictures until a new area located at the corresponding position is received. As another embodiment of our method, the area may be copied to subsequent pictures either until a new area located at the corresponding position is coded or until a maximum number of copies has been created. For example, this maximum number of copies may be signaled as part of the bitstream, may be defined in the specification text or may be derived in a different way.
2. FIG. 4 illustrates an example for the freeze process with four pictures at time instances t, t+1, t+2 and t+3. Each picture is divided into five slices denoted as “Slice 1” to “Slice 5”. It is worth noting that it is arbitrary that every picture is partitioned into the same number of slices and no requirement for the method. Additionally, each slice is labeled with the time instance at which it has been coded, e.g. “(t+1)” for a slice coded at time instance t+1. The copy procedure of slices for which the slice freeze process is used is highlighted by arrows.



FIG. 4 – Example for the slice freeze process with four pictures

1. Assuming that one or several areas are coded with the freeze process, it may be beneficial to introduce additional constraints for subsequent pictures, e.g. with respect to the partitioning of subsequent pictures into areas. Thus, as one additional embodiment it may be specified that the remaining part of the subsequent picture, which cannot be filled by copied freeze areas, must be completely filled by one or several areas which are coded for this subsequent picture. As another embodiment of our method, constraints may be specified for the case that the end of the freeze process for an area is indicated by coding a new area for the corresponding position in the picture. In such a case, it may be specified that the entire part of the picture previously covered by the freeze area must be covered by newly coded areas. Different constraints may be beneficial in other scenarios.
2. The usage of the described methods and the presence of the described syntax elements may be enabled and disabled. For instance, the methods and the presence of the corresponding syntax elements may be enabled by signaling syntax elements in the SPS, in the PPS, in the VPS or by other techniques. Examples for the signaling as part of the SPS syntax and as part of the PPS syntax are given in Table 2 and Table 3, respectively. Changes relative to the HEVC screen content coding draft, as described in R. Joshi and J. Xu, JCT-VC S1005: High Efficiency Video Coding (HEVC) Screen Content Coding: Draft 2, 19th Meeting of the Joint Collaborative Team on Video Coding (JCT-VC), Strasbourg, FR, 2014, are highlighted. In these examples, the syntax element “freeze\_process\_enabled\_flag” is used to signal the enabling of the freeze process. For instance, the freeze process may be enabled if freeze\_process\_enabled\_flag is equal to 1 while the freeze process may be disabled if freeze\_process\_enabled\_flag is equal to 0. Furthermore, the value of freeze\_process\_enabled flag may be inferred to 0 if the syntax element is not present in the bitstream. Other signaling techniques may be applied to enable or disable the freeze process.

|  |  |
| --- | --- |
| seq\_parameter\_set\_rbsp( ) { | Descriptor |
| **[…]** |  |
| **freeze\_process\_enabled\_flag** | u(1) |
| **[…]** |  |
| } |  |

Table 2 – Example for the freeze process enabling syntax as part of the sequence parameter set

|  |  |
| --- | --- |
| pic\_parameter\_set\_rbsp ( ) { | Descriptor |
| **[…]** |  |
| **freeze\_process\_enabled\_flag** | u(1) |
| **[…]** |  |
| } |  |

Table 3 – Example for the freeze process enabling syntax as part of the picture parameter set

1. As another embodiment of our method, the freeze process may be indicated using supplemental enhancement information (SEI) messages.
2. There may be a scenario in which only parts of a picture need to be decoded. For instance, this may be the case due to the usage of the previously described freeze process. In such a scenario it may be beneficial for the decoder to be aware in advance of the proportion of the picture parts which need to be decoded. The decoder may use this awareness for instance for a better resource allocation. This awareness may be generated for example by explicit signaling of the proportion as part of the bitstream, by analyzing previous pictures or by analyzing other syntax elements. For instance, the information may be signaled as part of the PPS syntax, as part of the SPS syntax, as part of the VPS syntax or as part of the slice segment header syntax among other possibilities. Table 4 gives an example for the signaling of the proportion of the picture which needs to be decoded. The syntax element *decode\_picture\_proportion* is used for this purpose.

|  |  |
| --- | --- |
| pic\_parameter\_set\_rbsp( ) { | Descriptor |
| **[…]** |  |
| **decode\_picture\_proportion** | ue(v) |
| **[…]** |  |
| } |  |

Table 4 - Example for the picture decoding proportion in the PPS syntax

1. A decoding process may be specified for the reconstruction of areas for which the freeze process is used. An example for the decoding process for a frozen slice segment is described in the following.
2. If the slice segment freeze mode is used for a slice segment, the following decoding process may be invoked for every color component to reconstruct the samples of the copied slice segment.
3. Inputs to this process are:

– a location ( xCurr, yCurr ) specifying the top-left sample of the frozen slice segment relative to the top‑left sample of the current picture component,

– an array recSamplesFrozenSlice specifying the reconstructed samples of the frozen slice segment,

– a variable cIdx specifying the color component.

1. Output of this process is an array recSamples specifying the reconstructed samples of the copied slice segment.
2. Depending on the value of the color component cIdx, the following assignments are made:

– If cIdx is equal to 0, recSamplesFrozenSlice corresponds to the reconstructed picture sample array after deblocking recPictureL of the picture to which the frozen slice corresponds and recSamples corresponds to the reconstructed picture sample array after deblocking recPictureL.

– Otherwise, if cIdx is equal to 1, recSamplesFrozenSlice corresponds to the reconstructed picture sample array after deblocking recPictureCb of the picture to which the frozen slice corresponds and recSamples corresponds to the reconstructed picture sample array after deblocking recPictureCb.

– Otherwise (cIdx is equal to 2), recSamplesFrozenSlice corresponds to the reconstructed picture sample array after deblocking recPictureCr of the picture to which the frozen slice corresponds and recSamples corresponds to the reconstructed picture sample array after deblocking recPictureCr.

1. The reconstructed samples array recSamples is derived as follows:

recSamples[ xCurr + i ][ yCurr + j ] = recSamplesFrozenSlice[ xCurr + i ][ yCurr + j ]

where i and j are used as control variables to increment the x and y coordinate in the sample array such that the locations ( xCurr + i, yCurr + j ) cover all locations in the frozen slice segment.

1. FIG. 5 is a schematic diagram of a network element 500 (e.g., a computer, server, smartphone, tablet computer, etc.) configured to implement the disclosed embodiments. Network element 500 comprises ports 510, transceiver units (Tx/Rx) 520, a processor 530, and a memory 540 comprising a coding module 550 (e.g., a static screen content coding module). Ports 510 are coupled to Tx/Rx 520, which may be transmitters, receivers, or combinations thereof. The Tx/Rx 520 may transmit and receive data via the ports 510. Processor 530 is configured to process data. Memory 540 is configured to store data and instructions for implementing embodiments described herein. The network element 500 may also comprise electrical-to-optical (EO) components and optical-to-electrical (OE) components coupled to the ports 510 and Tx/Rx 520 for receiving and transmitting electrical signals and optical signals.
2. The processor 530 may be implemented by hardware and software. The processor 530 may be implemented as one or more central processing unit (CPU) chips, logic units, cores (e.g., as a multi-core processor), field-programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), and digital signal processors (DSPs). The processor 530 is in communication with the ports 510, Tx/Rx 520, and memory 540.
3. The memory 540 comprises one or more of disks, tape drives, and solid-state drives and may be used as an over-flow data storage device, to store programs when such programs are selected for execution, and to store instructions and data that are read during program execution. The memory 540 may be volatile and non-volatile and may be read-only memory (ROM), random-access memory (RAM), ternary content-addressable memory (TCAM), and static random-access memory (SRAM). Coding module 550 is implemented by processor 530 to execute the instructions for implementing various embodiments previously discussed.

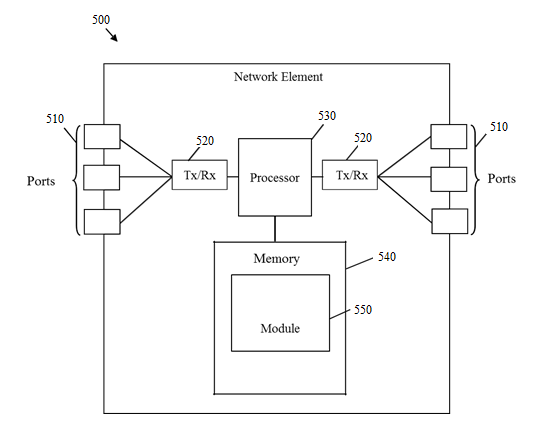


FIG. 5 - A schematic diagram of an embodiment of a network element

1. While several embodiments have been provided in the present disclosure, it should be understood that the disclosed systems and methods might be embodied in many other specific forms without departing from the spirit or scope of the present disclosure. The present examples are to be considered as illustrative and not restrictive, and the intention is not to be limited to the details given herein. For example, the various elements or components may be combined or integrated in another system or certain features may be omitted, or not implemented.
2. FIG. 6 illustrates an embodiment of a video encoder 605. The video encoder 605 may comprise a rate-distortion optimization (RDO) module 610, a prediction module 620, a transform module 630, a quantization module 640, an entropy encoder 650, a de-quantization module 660, an inverse transform module 670, and a reconstruction module 680 arranged as shown in FIG. 6. In operation, the video encoder 605 may receive an input video comprising a sequence of video frames (or slices). Herein, a frame may refer to any of a predicted frame (P-frame), an intra-coded frame (I-frame), or a bi-predictive frame (B-frame). Likewise, a slice may refer to any of a P-slice, an I-slice, or a B-slice.

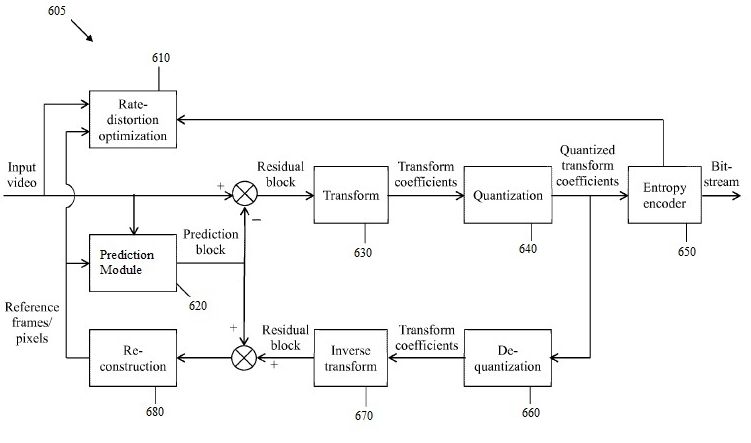


FIG. 6 – An encoder configured to implement the inventive concepts disclosed herein

1. The RDO module 610 may be configured to coordinate or make logic decisions for one or more of other modules. For example, based on one or more previously encoded frames, the RDO module 610 may determine how a current frame (or slice) being encoded is partitioned into a plurality of CUs, and how a CU is partitioned into one or more PUs and TUs. As noted above, CU, PU, and TU are various types of blocks used in HEVC. In addition, the RDO module 610 may determine how the current frame is to be predicted. The current frame may be predicted via inter and/or intra prediction. For intra prediction, there are a plurality of available prediction modes or directions in HEVC (e.g., 34 modes for the Y component and six modes (including LM mode) for the U or V component), and an optimal mode may be determined by the RDO module 610. For example, the RDO module 610 may calculate a sum of absolute error (SAE) for each prediction mode, and select a prediction mode that results in the smallest SAE.
2. In an embodiment, the prediction module 620 is configured to implement the inventive concepts disclosed herein to generate a prediction block for a current block from the input video. The prediction module 620 may utilize either reference frames for inter prediction or reference pixels in the current frame for intra prediction. The prediction block comprises a plurality of predicted pixel samples, each of which may be generated based on a plurality of reconstructed luma samples located in a corresponding reconstructed luma block, and a plurality of reconstructed chroma samples located in a corresponding reconstructed chroma block.
3. Upon generation of the prediction block for the current block, the current block may be subtracted by the prediction block, or vice versa, to generate a residual block. The residual block may be fed into the transform module 630, which may convert residual samples into a matrix of transform coefficients via a two-dimensional orthogonal transform, such as a discrete cosine transform (DCT). Then, the matrix of transform coefficients may be quantized by the quantization module 640 before being fed into the entropy encoder 650. The quantization module 640 may alter the scale of the transform coefficients and round them to integers, which may reduce the number of non-zero transform coefficients. As a result, a compression ratio may be increased. Quantized transform coefficients may be scanned and encoded by the entropy encoder 650 into an encoded bitstream. Further, to facilitate continuous encoding of blocks, the quantized transform coefficients may also be fed into the de-quantization module 660 to recover the original scale of the transform coefficients. Then, the inverse transform module 670 may perform the inverse of the transform module 630 and generate a noisy version of the original residual block. Then, the lossy residual block may be fed into the reconstruction module 680, which may generate reconstructed samples for intra prediction of future blocks. If desired, filtering may be performed on the reconstructed samples before they are used for intra prediction.
4. It should be noted that FIG. 6 may be a simplified illustration of a video encoder, thus it may include only part of modules present in the video encoder. Other modules (e.g., filter, scanner, and transmitter), although not shown in FIG. 6, may also be included to facilitate video encoding as understood by one of skill in the art. In addition, depending on the encoding scheme, some of the modules in the video encoder may be skipped. For example, in lossless encoding of certain video content, no information loss may be allowed, thus the quantization module 640 and the de-quantization module 660 may be skipped. For another example, if the residual block is encoded directly without being converted to transform coefficients, the transform module 630 and the inverse transform module 670 may be skipped. Moreover, prior to transmission from the encoder, the encoded bitstream may be configured to include other information, such as video resolution, frame rate, block partitioning information (sizes, coordinates), prediction modes, etc., so that the encoded sequence of video frames may be properly decoded by a video decoder.
5. FIG. 7 illustrates an embodiment of a video decoder 700. The video decoder 700 may correspond to the video encoder 605 of FIG. 6, and may comprise an entropy decoder 710, a de-quantization module 720, an inverse transform module 730, a prediction module 740, and a reconstruction module 750 arranged as shown in FIG. 7. In operation, an encoded bitstream containing information of a sequence of video frames may be received by the entropy decoder 710, which may decode the bitstream to an uncompressed format. A matrix of quantized transform coefficients may be generated, which may then be fed into the de-quantization module 720, which may be the same or similar to the de-quantization module 660 in FIG. 6. Then, output of the de-quantization module 720 may be fed into the inverse transform module 730, which may convert transform coefficients to residual values of a residual block. In addition, information containing a prediction mode of the current block may also be decoded by the entropy decoder 710. The prediction module 740 may generate a prediction block for the current block based on the inventive concepts disclosed herein.

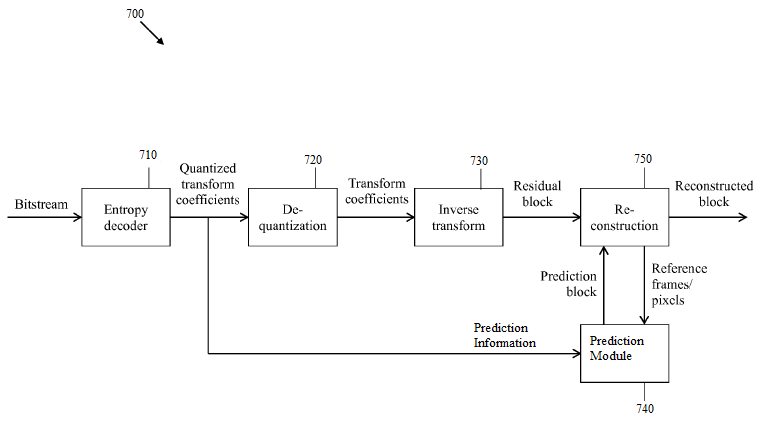


FIG. 7 – A decoder configured to implement the inventive concepts disclosed herein

1. In addition, techniques, systems, subsystems, and methods described and illustrated in the various embodiments as discrete or separate may be combined or integrated with other systems, modules, techniques, or methods without departing from the scope of the present disclosure. Other items shown or discussed as coupled or directly coupled or communicating with each other may be indirectly coupled or communicating through some interface, device, or intermediate component whether electrically, mechanically, or otherwise. Other examples of changes, substitutions, and alterations are ascertainable by one skilled in the art and could be made without departing from the spirit and scope disclosed herein.

**CLAIMS**

What is claimed is:

1. A system, a method, and an apparatus for implementing static screen content coding as shown and described herein.
2. The method of claim 1, wherein areas of a picture are frozen by applying the described freeze process.
3. The method of claim 2, wherein the frozen areas of a picture are copied to subsequently coded pictures.
4. The method of claim 2, wherein the freeze process is terminated by applying the described termination process.
5. The method of claim 1, wherein the described constraints with respect to the partitioning of pictures are applied.
6. The method of claim 1, wherein syntax elements corresponding to the freeze mode are signaled as part of the bitstream.
7. The method of claim 1, wherein the proportion of the picture, which needs to be decoded, is signaled as part of the bitstream.