(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 August 2014 (07.08.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/120960 A1

(51) **International Patent Classification**:
H04N 19/136 (2014.01) *H04N 21/2662* (2011.01)

(21) **International Application Number**:
PCT/US2014/013915

(22) **International Filing Date**:
30 January 2014 (30.01.2014)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
61/758,314 30 January 2013 (30.01.2013) US

(71) **Applicant** *(for all designated States except US)*: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95054 (US).

(72) **Inventors; and**
(71) **Applicants** *(for US only)*: **YENNETI, Sairam** [IN/IN]; D167, Second Floor Sector 26, Noida 201301 (IN). **PURI, Atul** [US/US]; 16080 NE 85th Street, Apt N204, Redmond, Washington 98052 (US).

(74) **Agent**: **BOOTZIN, Joel H.**; Lynch Law Patent Group PC, C/O CPA Global, P.O. Box 52050, Minneapolis, Minnesota 55402 (US).

(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) **Title**: CONTENT ADAPTIVE BITRATE AND QUALITY CONTROL BY USING FRAME HIERARCHY SENSITIVE QUANTIZATION FOR HIGH EFFICIENCY NEXT GENERATION VIDEO CODING

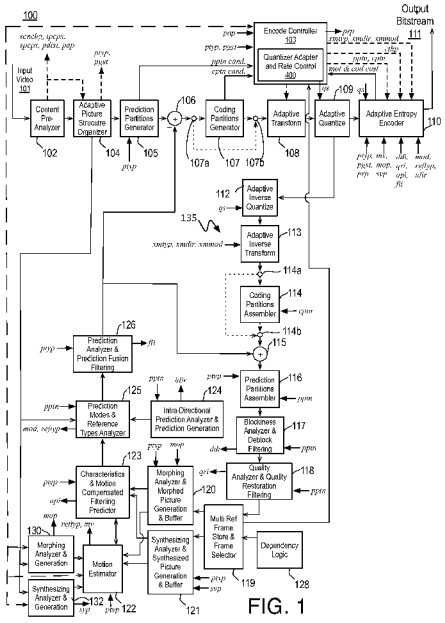


FIG. 1

(57) **Abstract**: This application relates to a computer-implemented method for high efficiency next generation video coding. A computer-implemented method for video coding, comprising: obtaining frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as reference, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both; and determining a quantization parameter for the frames depending at least on the level of the hierarchy of at least one current frame, and wherein each frame is given a rank associated with the level the frame is on.

WO 2014/120960 A1

Published:

— *with international search report (Art. 21(3))*

CONTENT ADAPTIVE BITRATE AND QUALITY CONTROL BY USING FRAME HIERARCHY SENSITIVE QUANTIZATION FOR HIGH EFFICIENCY NEXT GENERATION VIDEO CODING

RELATED APPLICATIONS

The present application claims the benefit of U.S. Provisional Application No. 61/758,314 filed 30 Jan 2013, and titled "NEXT GENERATION VIDEO CODING".

BACKGROUND

A video encoder compresses video information so that more information can be sent over a given bandwidth. The compressed signal may then be transmitted to a receiver having a decoder that decodes or decompresses the signal prior to display.

High Efficient Video Coding (HEVC) is the latest video compression standard, which is being developed by the Joint Collaborative Team on Video Coding (JCT-VC) formed by ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG). HEVC is being developed in response to the previous H.264/AVC (Advanced Video Coding) standard not providing enough compression for evolving higher resolution video applications. Similar to previous video coding standards, HEVC includes basic functional modules such as intra/inter prediction, transform, quantization, in-loop filtering, and entropy coding.

The ongoing HEVC standard may attempt to improve on limitations of the H.264/AVC standard such as limited choices for allowed prediction partitions and coding partitions, limited allowed multiple references and prediction generation, limited transform block sizes and actual transforms, limited mechanisms for reducing coding artifacts, and inefficient entropy encoding techniques. However, the ongoing HEVC standard may use iterative approaches to solving such problems.

For instance, with ever increasing resolution of video to be compressed and expectation of high video quality, the corresponding bitrate/bandwidth required for coding using existing video coding standards such as H.264 or even evolving standards such as H.265/HEVC, is relatively high. The aforementioned standards use expanded forms of traditional approaches to implicitly address the insufficient compression/quality problem, but often the results are limited.

The present description, developed within the context of a Next Generation Video (NGV) codec project, addresses the general problem of designing an advanced video codec that maximizes the achievable compression efficiency while remaining sufficiently practical for implementation on devices. For instance, with ever increasing resolution of video and expectation of high video quality due to availability of good displays, the corresponding bitrate/bandwidth required using existing video coding standards such as earlier MPEG standards and even the more recent H.264/AVC standard, is relatively high. H.264/AVC was not perceived to be providing high enough compression for evolving higher resolution video applications.

BRIEF DESCRIPTION OF THE DRAWINGS

The material described herein is illustrated by way of example and not by way of limitation in the accompanying figures. For simplicity and clarity of illustration, elements illustrated in the figures are not necessarily drawn to scale. For example, the dimensions of some elements may be exaggerated relative to other elements for clarity. Further, where considered appropriate, reference labels have been repeated among the figures to indicate corresponding or analogous elements. In the figures:

Fig. 1 is an illustrative diagram of an example next generation video encoder;

Fig. 2 is an illustrative diagram of an example next generation video decoder;

FIG. 3(a) is an illustrative diagram of an example next generation video encoder and subsystems;

FIG. 3(b) is an illustrative diagram of an example next generation video decoder and subsystems;

FIG. 4 is a diagram of a quantizer adapter rate control according to the disclosure herein;

FIG. 5 is a diagram of a two-pass quantizer adapter rate control according to the disclosure herein;

FIG. 6 is a flow chart of a quantizer adapter rate control process;

FIG. 7 is a flow chart of a one-pass quantizer adapter rate control process;

FIG. 8 is an illustrative diagram of a frame sequence with a hierarchical frame structure;

FIG. 9 is a flow chart of a rate control start up process;

FIG. 10 is an illustrative diagram of a hierarchical frame structure;

FIG. 12 is a chart showing Qp value versus complexity linear relation;

FIG. 13 is a chart showing Qp value versus complexity quadratic relation;

FIG. 14 is a table showing target bit rate bpp to bpp_id mapping;

FIG. 15 is a table showing complexity measure cpx to cpx_id mapping;

FIG. 16 is a table of mean Qp values;

FIG. 17 is a table of minimum Qp values;

FIG. 18 is a table of maximum Qp values;

FIG. 19 is a chart of I to P bits ratio versus Qp;

FIG. 20 is an illustrative diagram of a buffer;

FIG. 21 is a table of rate thresholds for buffer increments;

FIGS. 22A-22B is an illustrative flow chart of a two-pass quantizer adapter rate control process;

FIG. 23 is an illustrative flow chart of a one-pass quantizer rate control process with lookahead;

FIG. 24 is an illustrative diagram of an example video coding system;

FIG. 25 is an illustrative diagram of an example system; and

FIG. 26 illustrates an example device, all arranged in accordance with at least some implementations of the present disclosure.

DETAILED DESCRIPTION

One or more implementations are now described with reference to the enclosed figures. While specific configurations and arrangements are discussed, it should be understood that this is

done for illustrative purposes only. Persons skilled in the relevant art will recognize that other configurations and arrangements may be employed without departing from the spirit and scope of the description. It will be apparent to those skilled in the relevant art that techniques and/or arrangements described herein may also be employed in a variety of other systems and applications other than what is described herein.

While the following description sets forth various implementations that may be manifested in architectures such as system-on-a-chip (SoC) architectures for example, implementation of the techniques and/or arrangements described herein are not restricted to particular architectures and/or computing systems and may be implemented by any architecture and/or computing system for similar purposes. For instance, various architectures employing, for example, multiple integrated circuit (IC) chips and/or packages, and/or various computing devices and/or consumer electronic (CE) devices such as set top boxes, smart phones, etc., may implement the techniques and/or arrangements described herein. Further, while the following description may set forth numerous specific details such as logic implementations, types and interrelationships of system components, logic partitioning/integration choices, etc., claimed subject matter may be practiced without such specific details. In other instances, some material such as, for example, control structures and full software instruction sequences, may not be shown in detail in order not to obscure the material disclosed herein.

The material disclosed herein may be implemented in hardware, firmware, software, or any combination thereof. The material disclosed herein may also be implemented as instructions stored on a machine-readable medium, which may be read and executed by one or more processors. A machine-readable medium may include any medium and/or mechanism for storing or transmitting information in a form readable by a machine (e.g., a computing device). For example, a machine-readable medium may include read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; electrical, optical, acoustical or other forms of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.); and others.

References in the specification to "one implementation", "an implementation", "an example implementation", etc., indicate that the implementation described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same implementation. Further, when a particular feature, structure, or

characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to effect such feature, structure, or characteristic in connection with other implementations whether or not explicitly described herein.

As used herein, the term "coder" may refer to an encoder and/or a decoder. Similarly, as used herein, the term "coding" may refer to performing video encoding via an encoder and/or performing video decoding via a decoder. For example, a video encoder and video decoder may both be examples of coders capable of coding video data. In addition, as used herein, the term "codec" may refer to any process, program or set of operations, such as, for example, any combination of software, firmware, and/or hardware that may implement an encoder and/or a decoder. Further, as used herein, the phrase "video data" may refer to any type of data associated with video coding such as, for example, video frames, image data, encoded bit stream data, or the like.

Systems, apparatus, articles, and methods are described below related to content adaptive bitrate and quality control by quantization for high efficiency next generation video coding.

As discussed above, there are several reasons why the HEVC standard, while a good step forward, may not be sufficient for maximum compression efficiency while remaining sufficiently practical for implementation on devices. More specifically, a video encoder has a rate controller (RC) or bit-rate controller (BRC) that could be improved. Sometimes this component of a video encoder is standalone, while other times it is part of a larger component called the encode controller that manages many of the aspects of encoding including rate distortion, and that may manage overall bitrate when constant bit rate (CBR) coding is enabled.

Typical rate controller or bit-rate controller (components of the encode controller) used in conjunction with H.264/AVC video coding is very limited as the standard is not as bit-rate efficient as HEVC or that disclosed here. Since typically good rate controllers for a standard only become available when refined implementations of a standard become available, and since HEVC is relatively new, not many HEVC encoder implementations have good rate controllers. The aforementioned limitations of the state of the art standards and upcoming HEVC standard such as H.264 are addressed by the new and innovative rate controller described herein, and applied in context of NGV, but is also applicable to HEVC encoding.

NGV video coding differs from standards based approaches as it naturally incorporates significant content based adaptivity in video coding process to achieve higher compression. By

comparison, standards based video coding approaches typically tend to squeeze higher gains by adaptations and fine tuning of legacy approaches. For instance, all standards based approaches heavily rely on adapting and further tweaking of motion compensated interframe coding as the primary means to reduce prediction differences to achieve gains. On the other hand, NGV, in addition to exploiting interframe differences due to motion, also exploits other types of interframe differences (gain, blur, registration) that naturally exist in typical video scenes.

Herein, an encoder controller may have a quantization adapter and rate control that varies the quantization parameter Qp (also referred to as the quantizer) from frame to frame, in one example form, and depending on the level (also referred to as the rank herein) of the frame on a pyramid or hierarchical reference frame dependency structure on a sequence of frames that may be, or may be part of, a group of pictures (GOP), video, scene, or any other sequence of frames. This method and system obtains high compression without sacrificing high quality images. This quantization arrangement may be performed within a one-pass constant bitrate (CBR) rate control system that uses a buffer control and buffer model to modulate the Qp depending on the status or fullness of a buffer. Such a buffer by one example may be a virtual buffer at the encoder that imitates the size and behavior of a decoder buffer. Alternatively, a two-pass (or multi-pass) variable bitrate (VBR) system may be used that performs a first pass analysis on a frame sequence to determine which frames are more complex (or need special attention) and may need greater detail, and then at least a second pass analysis that changes the Qp of the frames for more efficient coding in light of the first pass analysis. This alternative may be used when sufficient time exists to analyze a frame sequence twice, such as with non-live (or non-real time) media formats for example. A third system, referred to as one-pass VBR rate control with lookahead, provides a compromise where the system merely looks a certain number of frames ahead to determine frame complexity instead of performing a full first pass analysis. These methodologies and systems are explained in detail below.

Referring to FIG. 1 for more detail, Fig. 1 is an illustrative diagram of an example next generation video encoder 100, arranged in accordance with at least some implementations of the present disclosure. As shown, encoder 100 may receive input video 101. Input video 101 may include any suitable input video for encoding such as, for example, input frames of a video sequence. As shown, input video 101 may be received via a content pre-analyzer module 102. Content pre-analyzer module 102 may be configured to perform analysis of the content of video frames of input video 101 to determine various types of parameters for improving video coding efficiency and speed performance. For example, content pre-analyzer module 102 may

determine horizontal and vertical gradient information (e.g., Rs, Cs), variance, spatial complexity per picture, temporal complexity per picture, scene change detection, motion range estimation, gain detection, prediction distance estimation, number of objects estimation, region boundary detection, spatial complexity map computation, focus estimation, film grain estimation, or the like. The parameters generated by content pre-analyzer module 102 may be used by encoder 100 (e.g., via encode controller 103) and/or quantized and communicated to a decoder. As shown, video frames and/or other data may be transmitted from content pre-analyzer module 102 to adaptive picture organizer module 104, which may determine the picture type (e.g., I-, P-, or F/B-picture) of each video frame and reorder the video frames as needed. In some examples, adaptive picture organizer module 104 may include a frame portion generator configured to generate frame portions. In some examples, content pre-analyzer module 102 and adaptive picture organizer module 104 may together be considered a pre-analyzer subsystem of encoder 100.

As shown, video frames and/or other data may be transmitted from adaptive picture organizer module 104 to prediction partitions generator module 105. In some examples, prediction partitions generator module 105 may divide a frame or picture into tiles or super-fragments or the like. In some examples, an additional module (e.g., between modules 104 and 105) may be provided for dividing a frame or picture into tiles or super-fragments. Prediction partitions generator module 105 may divide each tile or super-fragment into potential prediction partitionings or partitions. In some examples, the potential prediction partitionings may be determined using a partitioning technique such as, for example, a k-d tree partitioning technique, a bi-tree partitioning technique, or the like, which may be determined based on the picture type (e.g., I-, P-, or F/B-picture) of individual video frames, a characteristic of the frame portion being partitioned, or the like. In some examples, the determined potential prediction partitionings may be partitions for prediction (e.g., inter- or intra-prediction) and may be described as prediction partitions or prediction blocks or the like.

In some examples, a selected prediction partitioning (e.g., prediction partitions) may be determined from the potential prediction partitionings. For example, the selected prediction partitioning may be based on determining, for each potential prediction partitioning, predictions using characteristics and motion based multi-reference predictions or intra-predictions, and determining prediction parameters. For each potential prediction partitioning, a potential prediction error may be determined by differencing original pixels with prediction pixels and the selected prediction partitioning may be the potential prediction partitioning with the minimum

prediction error. In other examples, the selected prediction partitioning may be determined based on a rate distortion optimization including a weighted scoring based on number of bits for coding the partitioning and a prediction error associated with the prediction partitioning.

As shown, the original pixels of the selected prediction partitioning (e.g., prediction partitions of a current frame) may be differenced with predicted partitions (e.g., a prediction of the prediction partition of the current frame based on a reference frame or frames and other predictive data such as inter- or intra-prediction data) at differencer 106. The determination of the predicted partitions will be described further below and may include a decode loop 135 as shown in Fig. 1. Any residuals or residual data (e.g., partition prediction error data) from the differencing may be transmitted to coding partitions generator module 107. In some examples, such as for intra-prediction of prediction partitions in any picture type (I-, F/B- or P-pictures), coding partitions generator module 107 may be bypassed via switches 107a and 107b. In such examples, only a single level of partitioning may be performed. Such partitioning may be described as prediction partitioning (as discussed) or coding partitioning or both. In various examples, such partitioning may be performed via prediction partitions generator module 105 (as discussed) or, as is discussed further herein, such partitioning may be performed via a k-d tree intra-prediction/coding partitioner module or a bi-tree intra-prediction/coding partitioner module implemented via coding partitions generator module 107.

In some examples, the partition prediction error data, if any, may not be significant enough to warrant encoding. In other examples, where it may be desirable to encode the partition prediction error data and the partition prediction error data is associated with inter-prediction or the like, coding partitions generator module 107 may determine coding partitions of the prediction partitions. In some examples, coding partitions generator module 107 may not be needed as the partition may be encoded without coding partitioning (e.g., as shown via the bypass path available via switches 107a and 107b). With or without coding partitioning, the partition prediction error data (which may subsequently be described as coding partitions in either event) may be transmitted to adaptive transform module 108 in the event the residuals or residual data require encoding. In some examples, prediction partitions generator module 105 and coding partitions generator module 107 may together be considered a partitioner subsystem of encoder 100. In various examples, coding partitions generator module 107 may operate on partition prediction error data, original pixel data, residual data, or wavelet data.

Coding partitions generator module 107 may generate potential coding partitionings (e.g., coding partitions) of, for example, partition prediction error data using bi-tree and/or k-d tree partitioning techniques or the like. In some examples, the potential coding partitions may be transformed using adaptive or fixed transforms with various block sizes via adaptive transform module 108 and a selected coding partitioning and selected transforms (e.g., adaptive or fixed) may be determined based on a rate distortion optimization or other basis. In some examples, the selected coding partitioning and/or the selected transform(s) may be determined based on a predetermined selection method based on coding partitions size or the like.

For example, adaptive transform module 108 may include a first portion or component for performing a parametric transform to allow locally optimal transform coding of small to medium size blocks and a second portion or component for performing globally stable, low overhead transform coding using a fixed transform, such as a discrete cosine transform (DCT) or a picture based transform from a variety of transforms, including parametric transforms, or any other configuration as is discussed further herein. In some examples, for locally optimal transform coding a Parametric Haar Transform (PHT) may be performed, as is discussed further herein. In some examples, transforms may be performed on 2D blocks of rectangular sizes between about 4x4 pixels and 64x64 pixels, with actual sizes depending on a number of factors such as whether the transformed data is luma or chroma, or inter or intra, or if the determined transform used is PHT or DCT or the like.

As shown, the resultant transform coefficients may be transmitted to adaptive quantize module 109. Adaptive quantize module 109 may quantize the resultant transform coefficients. Further, any data associated with a parametric transform, as needed, may be transmitted to either adaptive quantize module 109 (if quantization is desired) or adaptive entropy encoder module 110. Also as shown in FIG. 1, the quantized coefficients may be scanned and transmitted to adaptive entropy encoder module 110. Adaptive entropy encoder module 110 may entropy encode the quantized coefficients and include them in output bitstream 111. In some examples, adaptive transform module 108 and adaptive quantize module 109 may together be considered a transform encoder subsystem of encoder 100. The encode controller 103 may have a quantizer adapter and rate control 400 or 500 to determine and provide the quantization information set (qs) to the adaptive quantize module 109, and which may include the quantizer or quantizer parameters (Qp) as described in greater detail below, and the quantizer matrix (QM) choices

As also shown in FIG. 1, encoder 100 includes the local decode loop 135. The local decode loop 135 may begin at adaptive inverse quantize module 112. Adaptive inverse quantize module 112 may be configured to perform the opposite operation(s) of adaptive quantize module 109 such that an inverse scan may be performed and quantized coefficients may be de-scaled to determine transform coefficients. Such an adaptive quantize operation may be lossy, for example. As shown, the transform coefficients may be transmitted to an adaptive inverse transform module 113. Adaptive inverse transform module 113 may perform the inverse transform as that performed by adaptive transform module 108, for example, to generate residuals or residual values or partition prediction error data (or original data or wavelet data, as discussed) associated with coding partitions. In some examples, adaptive inverse quantize module 112 and adaptive inverse transform module 113 may together be considered a transform decoder subsystem of encoder 100.

As shown, the partition prediction error data (or the like) may be transmitted to optional coding partitions assembler 114. Coding partitions assembler 114 may assemble coding partitions into decoded prediction partitions as needed (as shown, in some examples, coding partitions assembler 114 may be skipped via switches 114a and 114b such that decoded prediction partitions may have been generated at adaptive inverse transform module 113) to generate prediction partitions of prediction error data or decoded residual prediction partitions or the like.

As shown, the decoded residual prediction partitions may be added to predicted partitions (e.g., prediction pixel data) at adder 115 to generate reconstructed prediction partitions. The reconstructed prediction partitions may be transmitted to prediction partitions assembler 116. Prediction partitions assembler 116 may assemble the reconstructed prediction partitions to generate reconstructed tiles or super-fragments. In some examples, coding partitions assembler module 114 and prediction partitions assembler module 116 may together be considered an un-partitioner subsystem of encoder 100.

The reconstructed tiles or super-fragments may be transmitted to blockiness analyzer and deblock filtering module 117. Blockiness analyzer and deblock filtering module 117 may deblock and dither the reconstructed tiles or super-fragments (or prediction partitions of tiles or super-fragments). The generated deblock and dither filter parameters may be used for the current filter operation and/or coded in output bitstream 111 for use by a decoder, for example. The output of blockiness analyzer and deblock filtering module 117 may be transmitted to a quality

analyzer and quality restoration filtering module 118. Quality analyzer and quality restoration filtering module 118 may determine QR filtering parameters (e.g., for a QR decomposition) and use the determined parameters for filtering. The QR filtering parameters may also be coded in output bitstream 111 for use by a decoder. As shown, the output of quality analyzer and quality restoration filtering module 118 may be transmitted to a multi-reference frame storage and frame selector (or multi reference control or decoded picture buffer or storage) 119. In some examples, the output of quality analyzer and quality restoration filtering module 118 may be a final reconstructed frame that may be used for prediction for coding other frames (e.g., the final reconstructed frame may be a reference frame or the like). In some examples, blockiness analyzer and deblock filtering module 117 and quality analyzer and quality restoration filtering module 118 may together be considered a filtering subsystem of encoder 100.

A dependency logic module 128 may provide indices for listing the reference frames and the relationship among the frames such as frame dependencies, or more specifically partition dependencies, for proper ordering and use for the frames by the multi reference control 119, and when certain frames are to be selected for prediction of another frame. This may include providing the dependency logic for picture group structures such as multi-reference prediction, chain prediction, hierarchal structures, and/or other prediction techniques

In encoder 100, prediction operations may include inter- and/or intra-prediction. As shown in FIG. 1(a), inter-prediction may be performed by one or more modules including morphing analyzer and morphed picture generation and buffer module 120 (or in-loop morphing generation module), synthesizing analyzer and synthesized picture generation and buffer module 121 (or in-loop synthesizing generation module), characteristics and motion filtering predictor module 123, morphing analyzer and generation module (or out-of-loop morphing analyzer module) 130, and synthesizing analyzer and generation module (or out-of-loop synthesizing analyzer module) 132, where the morphing and synthesis generators 120 and 121 are considered in-loop (in the decoder loop of the encoder), and where the morphing and synthesis analyzers 130 and 132 are considered out-of-loop (out of the decoder loop at the encoder). Note that while one is called an analyzer and the other a generator, both in-loop and out-of-loop modules may perform the same or similar tasks (forming modified frames and modification parameters for morphing and/or synthesis). Using these components, morphing generation module 120, or morphing analyzer 130, may permit various forms of morphing and may analyze a current picture to determine morphing parameters for (1) changes in gain, (2) changes in dominant (or global) motion, (3)

changes in registration, and/or (4) changes in blur with respect to a reference frame or frames with which it is to be coded, and prior to motion compensated prediction.

The out-of-loop morphing analyzer 130 and the synthesizing analyzer 132 receive picture group structure data from the adaptive picture organizer 104 and communicate with the encoder controller 103 to form the morphing and synthesis parameters (*mop, syp*) and modified reference frames based on the non-quantized, non-decoded, original frame data. The formation of the modified reference frames and modification parameters from the out-of-loop morphing and synthesis analyzers 130 and 132 may be much faster than that provided through the decoder loop 135, and this is especially advantageous for real time encoding. However, the use of the modified frames and parameters to perform compensation at another location, such as by a decoder, should be performed by the in-loop morphing and synthesis generators 120 and 121 on the decoding loop side of the encoder so that the correct compensation can be repeated when reconstructing frames at the decoder. Thus, the resulting modification parameters from the out-of-loop analyzers 130 and 132 are used by the in-loop morphing and synthesizing generator 120 and 121 to form the modified reference frames and for motion estimation by the motion estimator 122 to compute motion vectors. Thus, the computed morphing and synthesis parameters (*mop and syp*) may be quantized/de-quantized and used (for example, by morphing generation module 120) to generate morphed reference frames that may be used by motion estimator module 122 for computing motion vectors for efficient motion (and characteristics) compensated prediction of a current frame. The synthesizing generation module 121 uses several types of synthesized frames including super resolution (SR) pictures, projected interpolation (PI) pictures, among others in which motion compensated prediction can result in even higher gains by determining motion vectors for efficient motion compensated prediction in these frames. The details for some examples to perform morphing or synthesis are provided below

Motion estimator module 122 may generate motion vector data based on morphed reference frame(s) and/or super resolution (SR) pictures and projected interpolation (PI) pictures along with the current frame. In some examples, motion estimator module 122 may be considered an inter-prediction module. For example, the motion vector data may be used for inter-prediction. If inter-prediction is applied, characteristics and motion compensated filtering predictor module 123 may apply motion compensation as part of the local decode loop as discussed.

Intra-prediction may be performed by intra-directional prediction analyzer and prediction generation module 124. Intra-directional prediction analyzer and prediction generation module 124 may be configured to perform spatial directional prediction and may use decoded neighboring partitions. In some examples, both the determination of direction and generation of prediction may be performed by intra-directional prediction analyzer and prediction generation module 124. In some examples, intra-directional prediction analyzer and prediction generation module 124 may be considered an intra-prediction module.

As shown in FIG. 1, prediction modes and reference types analyzer module 125 may allow for selection of prediction modes from among, "skip", "auto", "inter", "split", "multi", and "intra", for each prediction partition of a tile (or super-fragment), all of which may apply to P- and F/B-pictures. In addition to prediction modes, it also allows for selection of reference types that can be different depending on "inter" or "multi" mode, as well as for P- and F/B-pictures. The prediction signal at the output of prediction modes and reference types analyzer module 125 may be filtered by prediction analyzer and prediction fusion filtering module 126. Prediction analyzer and prediction fusion filtering module 126 may determine parameters (e.g., filtering coefficients, frequency, overhead) to use for filtering and may perform the filtering. In some examples, filtering the prediction signal may fuse different types of signals representing different modes (e.g., intra, inter, multi, split, skip, and auto). In some examples, intra-prediction signals may be different than all other types of inter-prediction signal(s) such that proper filtering may greatly enhance coding efficiency. In some examples, the filtering parameters may be encoded in output bitstream 111 for use by a decoder. The filtered prediction signal may provide the second input (e.g., prediction partition(s)) to differencer 106, as discussed above, that may determine the prediction difference signal (e.g., partition prediction error) for coding discussed earlier. Further, the same filtered prediction signal may provide the second input to adder 115, also as discussed above. As discussed, output bitstream 111 may provide an efficiently encoded bitstream for use by a decoder for the presentment of video.

In operation, some components of encoder 100 may operate as an encoder prediction subsystem. For example, such an encoder prediction subsystem of encoder 100 may include multi-reference frame storage and frame selector 119, in-loop morphing analyzer and generation module 120, in-loop synthesizing analyzer and generation module 121, motion estimator module 122, and/or characteristics and motion compensated precision adaptive filtering predictor module 123 as well as out-of-loop morphing analyzer 130 and synthesizing analyzer 132.

In some implementations, such an encoder prediction subsystem of encoder 100 may incorporate a number of components and the combined predictions generated by these components in an efficient video coding algorithm. For example, proposed implementation of the NGV coder may include one or more of the following features: 1. Gain Compensation (e.g., explicit compensation for changes in gain/brightness in a scene); 2. Blur Compensation: e.g., explicit compensation for changes in blur/sharpness in a scene; 3. Dominant/Global Motion Compensation (e.g., explicit compensation for dominant motion in a scene); 4. Registration Compensation (e.g., explicit compensation for registration mismatches in a scene); 5. Super Resolution (e.g., explicit model for changes in resolution precision in a scene); 6. Projection (e.g., explicit model for changes in motion trajectory in a scene); the like, and/or combinations thereof.

FIG. 1 illustrates example control signals associated with operation of video encoder 100, where the following abbreviations may represent the associated information:

scnchg	Scene change information
spcpx	Spatial complexity information
tpcpx	Temporal complexity information
pdist	Temporal prediction distance information
pap	Pre Analysis parameters (placeholder for all other pre analysis parameters except scnchg, spcpx, tpcpx, pdist)
ptyp	Picture types information
pgst	Picture group structure information
pptn cand.	Prediction partitioning candidates
cptn cand.	Coding Partitioning Candidates
prp	Preprocessing
xmtyp	Transform type information
xmdir	Transform direction information
xmmod	Transform mode
ethp	One eighth (1/8th) pel motion prediction
pptn	Prediction Partitioning

cptn	Coding Partitioning
mot&cod cost	Motion and Coding Cost
qs	quantizer information set (includes Quantizer parameter (Qp), Quantizer matrix (QM) choice)
mv	Motion vectors
mop	Morphing Paramters
syp	Synthesizing Parameters
ddi	Deblock and dither information
qri	Quality Restoration filtering index/information
api	Adaptive Precision filtering index/information
fii	Fusion Filtering index/information
mod	Mode information
reftyp	Reference type information
idir	Intra Prediction Direction

The various signals and data items that may need to be sent to the decoder, ie, *pgst*, *ptyp*, *prp*, *pptn*, *cptn*, *modes*, *reftype*, *ethp*, *xmtyp*, *xmdir*, *xmmod*, *idir*, *mv*, *qs*, *mop*, *syp*, *ddi*, *qri*, *api*, *fii*, quant coefficients and others may then be entropy encoded by adaptive entropy encoder 110 that may include different entropy coders collectively referred to as an entropy encoder subsystem. While these control signals are illustrated as being associated with specific example functional modules of encoder 100 in Fig. 1, other implementations may include a different distribution of control signals among the functional modules of encoder 300. The present disclosure is not limited in this regard and, in various examples, implementation of the control signals herein may include the undertaking of only a subset of the specific example control signals shown, additional control signals, and/or in a different arrangement than illustrated.

FIG. 2 is an illustrative diagram of an example next generation video decoder 200, arranged in accordance with at least some implementations of the present disclosure. As shown, decoder 200 may receive an input bitstream 201. In some examples, input bitstream 201 may be encoded via encoder 100 and/or via the encoding techniques discussed herein. As shown, input bitstream 201 may be received by an adaptive entropy decoder module 202. Adaptive entropy decoder module 202 may decode the various types of encoded data (e.g., overhead, motion vectors, transform coefficients, etc.). In some examples, adaptive entropy decoder 202 may use a

variable length decoding technique. In some examples, adaptive entropy decoder 202 may perform the inverse operation(s) of adaptive entropy encoder module 110 discussed above.

The decoded data may be transmitted to adaptive inverse quantize module 203. Adaptive inverse quantize module 203 may be configured to inverse scan and de-scale quantized coefficients to determine transform coefficients. Such an adaptive quantize operation may be lossy, for example. In some examples, adaptive inverse quantize module 203 may be configured to perform the opposite operation of adaptive quantize module 109 (e.g., substantially the same operations as adaptive inverse quantize module 112). As shown, the transform coefficients (and, in some examples, transform data for use in a parametric transform) may be transmitted to an adaptive inverse transform module 204. Adaptive inverse transform module 204 may perform an inverse transform on the transform coefficients to generate residuals or residual values or partition prediction error data (or original data or wavelet data) associated with coding partitions. In some examples, adaptive inverse transform module 204 may be configured to perform the opposite operation of adaptive transform module 108 (e.g., substantially the same operations as adaptive inverse transform module 113). In some examples, adaptive inverse transform module 204 may perform an inverse transform based on other previously decoded data, such as, for example, decoded neighboring partitions. In some examples, adaptive inverse quantize module 203 and adaptive inverse transform module 204 may together be considered a transform decoder subsystem of decoder 200.

As shown, the residuals or residual values or partition prediction error data may be transmitted to coding partitions assembler 205. Coding partitions assembler 205 may assemble coding partitions into decoded prediction partitions as needed (as shown, in some examples, coding partitions assembler 205 may be skipped via switches 205a and 205b such that decoded prediction partitions may have been generated at adaptive inverse transform module 204). The decoded prediction partitions of prediction error data (e.g., prediction partition residuals) may be added to predicted partitions (e.g., prediction pixel data) at adder 206 to generate reconstructed prediction partitions. The reconstructed prediction partitions may be transmitted to prediction partitions assembler 207. Prediction partitions assembler 207 may assemble the reconstructed prediction partitions to generate reconstructed tiles or super-fragments. In some examples, coding partitions assembler module 205 and prediction partitions assembler module 207 may together be considered an un-partitioner subsystem of decoder 200.

The reconstructed tiles or super-fragments may be transmitted to deblock filtering module 208. Deblock filtering module 208 may deblock and dither the reconstructed tiles or super-fragments (or prediction partitions of tiles or super-fragments). The generated deblock and dither filter parameters may be determined from input bitstream 201, for example. The output of deblock filtering module 208 may be transmitted to a quality restoration filtering module 209. Quality restoration filtering module 209 may apply quality filtering based on QR parameters, which may be determined from input bitstream 201, for example. In some examples, deblock filtering module 208 and quality restoration filtering module 209 may together be considered a filtering subsystem of decoder 200.

As shown in FIG. 2, the output of quality restoration filtering module 209 may be final recon frames, and may be saved in multi-reference frame storage and frame selector (also may be called decoded picture buffer) 210, and are used (or morphed) to create morphed pictures/local buffers (at morphed picture generator and buffer 211) depending on the applied, decoded *mop* parameters. Likewise synthesized picture and local buffers (at synthesized picture generation and buffer 212) are created by applying decoded syp parameters to multi-reference frame storage and frame selector 210 (or in other words, the reconstructed frames in the storage or buffer 210). A dependency logic 220 may hold the index for, and perform the indexing for, the stored frames in the multi-reference frame storage 210. The indexing may be used for prediction techniques such as multi-reference frames, chain prediction and/or hierarchal (or pyramid) frame structures, and/or others as described below. The morphed and synthesized frames are used for motion compensated prediction that uses adaptive precision (AP) filtering based on *api* parameters, and keeps either 1/4 or 1/8 pel prediction depending on a decoded *ethp* signal.

As discussed, compensation due to prediction operations may include inter- and/or intra-prediction compensation. As shown, inter-prediction compensation may be performed by one or more modules including morphed picture generation and buffer module 211, synthesized picture generation and buffer module 212, and characteristics and motion compensated filtering predictor module 213. Morphing generation module 211 may use de-quantized morphing parameters (e.g., determined from input bitstream 201) to generate morphed reference frames. Synthesizing generation module 212 may generate super resolution (SR) pictures and projected interpolation (PI) pictures or the like based on parameters determined from input bitstream 201. If inter-prediction is applied, characteristics and motion compensated filtering predictor module

213 may apply motion compensation based on the received frames and motion vector data or the like in input bitstream 201.

Intra-prediction compensation may be performed by intra-directional prediction generation module 214. Intra-directional prediction generation module 214 may be configured to perform spatial directional prediction and may use decoded neighboring partitions according to intra-prediction data in input bitstream 201.

As shown in FIG. 2, prediction modes selector module 215 may determine a prediction mode selection from among, "skip", "auto", "inter", "multi", and "intra", for each prediction partition of a tile, all of which may apply to P- and F/B-pictures, based on mode selection data in input bitstream 201. In addition to prediction modes, it also allows for selection of reference types that can be different depending on "inter" or "multi" mode, as well as for P- and F/B-pictures. The prediction signal at the output of prediction modes selector module 215 may be filtered by prediction fusion filtering module 216. Prediction fusion filtering module 216 may perform filtering based on parameters (e.g., filtering coefficients, frequency, overhead) determined via input bitstream 201. In some examples, filtering the prediction signal may fuse different types of signals representing different modes (e.g., intra, inter, multi, skip, and auto). In some examples, intra-prediction signals may be different than all other types of inter-prediction signal(s) such that proper filtering may greatly enhance coding efficiency. The filtered prediction signal may provide the second input (e.g., prediction partition(s)) to differencer 206, as discussed above.

As discussed, the output of quality restoration filtering module 209 may be a final reconstructed frame. Final reconstructed frames may be transmitted to an adaptive picture re-organizer 217, which may re-order or re-organize frames as needed based on ordering parameters in input bitstream 201. Re-ordered frames may be transmitted to content post-restorer module 218. Content post-restorer module 218 may be an optional module configured to perform further improvement of perceptual quality of the decoded video. The improvement processing may be performed in response to quality improvement parameters in input bitstream 201 or it may be performed as standalone operation. In some examples, content post-restorer module 218 may apply parameters to improve quality such as, for example, an estimation of film grain noise or residual blockiness reduction (e.g., even after the deblocking operations discussed with respect to deblock filtering module 208). As shown, decoder 200 may provide display video 219, which may be configured for display via a display device (not shown).

FIG. 2 illustrates example control signals associated with operation of video decoder 200, where the indicated abbreviations may represent similar information as discussed with respect to FIG. 1 above. While these control signals are illustrated as being associated with specific example functional modules of decoder 200 in Fig. 4, other implementations may include a different distribution of control signals among the functional modules of encoder 100. The present disclosure is not limited in this regard and, in various examples, implementation of the control signals herein may include the undertaking of only a subset of the specific example control signals shown, additional control signals, and/or in a different arrangement than illustrated.

While FIGS. 1 through 2 illustrate particular encoding and decoding modules, various other coding modules or components not depicted may also be utilized in accordance with the present disclosure. Further, the present disclosure is not limited to the particular components illustrated in FIGS. 1 and 2 and/or to the manner in which the various components are arranged. Various components of the systems described herein may be implemented in software, firmware, and/or hardware and/or any combination thereof. For example, various components of encoder 100 and/or decoder 200 may be provided, at least in part, by hardware of a computing System-on-a-Chip (SoC) such as may be found in a computing system such as, for example, a mobile phone.

Further, it may be recognized that encoder 100 may be associated with and/or provided by a content provider system including, for example, a video content server system, and that output bitstream 111 may be transmitted or conveyed to decoders such as, for example, decoder 200 by various communications components and/or systems such as transceivers, antennae, network systems, and the like not depicted in FIGS. 1 and 2. It may also be recognized that decoder 200 may be associated with a client system such as a computing device (e.g., a desktop computer, laptop computer, tablet computer, convertible laptop, mobile phone, or the like) that is remote to encoder 100 and that receives input bitstream 201 via various communications components and/or systems such as transceivers, antennae, network systems, and the like not depicted in FIGS. 1 and 2. Therefore, in various implementations, encoder 100 and decoder subsystem 200 may be implemented either together or independent of one another.

FIG. 3(a) is an illustrative diagram of an example next generation video encoder 300a, arranged in accordance with at least some implementations of the present disclosure. FIG. 3(a) presents a similar encoder to that shown in FIGS. 1(a) and 1(b), and similar elements will not be

repeated for the sake of brevity. As shown in FIG. 3(a), encoder 300a may include preanalyzer subsystem 310a, partitioner subsystem 320a, prediction encoding subsystem 330a, transform encoder subsystem 340a, filtering encoding subsystem 350a, entropy encoder system 360a, transform decoder subsystem 370a, and/or unpartitioner subsystem 380a. Preanalyzer subsystem 310a may include content pre-analyzer module 102 and/or adaptive picture organizer module 104. Partitioner subsystem 320a may include prediction partitions generator module 105, and/or coding partitions generator 107. Prediction encoding subsystem 330a may include motion estimator module 122, characteristics and motion compensated filtering predictor module 123, and/or intra-directional prediction analyzer and prediction generation module 124. Transform encoder subsystem 340a may include adaptive transform module 108 and/or adaptive quantize module 109. Filtering encoding subsystem 350a may include blockiness analyzer and deblock filtering module 117, quality analyzer and quality restoration filtering module 118, motion estimator module 122, characteristics and motion compensated filtering predictor module 123, and/or prediction analyzer and prediction fusion filtering module 126. Entropy coding subsystem 360a may include adaptive entropy encoder module 110. Transform decoder subsystem 370a may include adaptive inverse quantize module 112 and/or adaptive inverse transform module 113. Unpartitioner subsystem 380a may include coding partitions assembler 114 and/or prediction partitions assembler 116.

Partitioner subsystem 320a of encoder 300a may include two partitioning subsystems: prediction partitions generator module 105 that may perform analysis and partitioning for prediction, and coding partitions generator module 107 that may perform analysis and partitioning for coding. Another partitioning method may include adaptive picture organizer 104 which may segment pictures into regions or slices may also be optionally considered as being part of this partitioner.

Prediction encoder subsystem 330a of encoder 300a may include motion estimator 122 and characteristics and motion compensated filtering predictor 123 that may perform analysis and prediction of “inter” signal, and intra-directional prediction analyzer and prediction generation module 124 that may perform analysis and prediction of “intra” signal. Motion estimator 122 and characteristics and motion compensated filtering predictor 123 may allow for increasing predictability by first compensating for other sources of differences (such as gain, global motion, registration), followed by actual motion compensation. They may also allow for use of data modeling to create synthesized frames (super resolution, and projection) that may allow better predictions, followed by use of actual motion compensation in such frames.

Transform encoder subsystem 340a of encoder 300a may perform analysis to select the type and size of transform and may include two major types of components. The first type of component may allow for using parametric transform to allow locally optimal transform coding of small to medium size blocks; such coding however may require some overhead. The second type of component may allow globally stable, low overhead coding using a generic/fixed transform such as the DCT, or a picture based transform from a choice of small number of transforms including parametric transforms. For locally adaptive transform coding, PHT (Parametric Haar Transform) may be used. Transforms may be performed on 2D blocks of rectangular sizes between 4x4 and 64x64, with actual sizes that may depend on a number of factors such as if the transformed data is luma or chroma, inter or intra, and if the transform used is PHT or DCT. The resulting transform coefficients may be quantized, scanned and entropy coded.

Entropy encoder subsystem 360a of encoder 300a may include a number of efficient but low complexity components each with the goal of efficiently coding a specific type of data (various types of overhead, motion vectors, or transform coefficients). Components of this subsystem may belong to a generic class of low complexity variable length coding techniques, however, for efficient coding, each component may be custom optimized for highest efficiency. For instance, a custom solution may be designed for coding of "Coded/Not Coded" data, another for "Modes and Ref Types" data, yet another for "Motion Vector" data, and yet another one for "Prediction and Coding Partitions" data. Finally, because a very large portion of data to be entropy coded is "transform coefficient" data, multiple approaches for efficient handling of specific block sizes, as well as an algorithm that may adapt between multiple tables may be used.

Filtering encoder subsystem 350a of encoder 300a may perform analysis of parameters as well as multiple filtering of the reconstructed pictures based on these parameters, and may include several subsystems. For example, a first subsystem, blockiness analyzer and deblock filtering module 117 may deblock and dither to reduce or mask any potential block coding artifacts. A second example subsystem, quality analyzer and quality restoration filtering module 118, may perform general quality restoration to reduce the artifacts due to quantization operation in any video coding. A third example subsystem, which may include motion estimator 122 and characteristics and motion compensated filtering predictor module 123, may improve results from motion compensation by using a filter that adapts to the motion characteristics (motion speed/degree of blurriness) of the content. A fourth example subsystem, prediction fusion analyzer and filter generation module 126, may allow adaptive filtering of the prediction signal

(which may reduce spurious artifacts in prediction, often from intra prediction) thereby reducing the prediction error which needs to be coded.

Encode controller module 103 of encoder 300a may be responsible for overall video quality under the constraints of given resources and desired encoding speed. For instance, in full RDO (Rate Distortion Optimization) based coding without using any shortcuts, the encoding speed for software encoding may be simply a consequence of computing resources (speed of processor, number of processors, hyperthreading, DDR3 memory etc.) availability. In such case, encode controller module 103 may be input every single combination of prediction partitions and coding partitions and by actual encoding, and the bitrate may be calculated along with reconstructed error for each case and, based on lagrangian optimization equations and in in compliance with the implementations described below, the best set of prediction and coding partitions may be sent for each tile of each frame being coded. The full RDO based mode may result in best compression efficiency and may also be the slowest encoding mode. By using content analysis parameters from content preanalyzer module 102 and using them to make RDO simplification (not test all possible cases) or only pass a certain percentage of the blocks through full RDO, quality versus speed tradeoffs may be made allowing speedier encoding. Encode controller module 103 may also include a quantizer adapter and rate control 400 or 500 described in detail below and that can be invoked to apply the implementations disclosed herein. While variable bitrate (VBR) based encoder operation has been mentioned, additional constant bitrate (CBR) controlled coding may be used.

Lastly, preanalyzer subsystem 310a of encoder 300a may perform analysis of content to compute various types of parameters useful for improving video coding efficiency and speed performance. For instance, it may compute horizontal and vertical gradient information (Rs, Cs), variance, spatial complexity per picture, temporal complexity per picture, scene change detection, motion range estimation, gain detection, prediction distance estimation, number of objects estimation, region boundary detection, spatial complexity map computation, focus estimation, film grain estimation etc. The parameters generated by preanalyzer subsystem 310a may either be consumed by the encoder or be quantized and communicated to decoder 200.

While subsystems 310a through 380a are illustrated as being associated with specific example functional modules of encoder 300a in Fig. 3(a), other implementations of encoder 300a herein may include a different distribution of the functional modules of encoder 300a among subsystems 310a through 380a. The present disclosure is not limited in this regard and, in various examples, implementation of the example subsystems 310a through 380a herein may

include the undertaking of only a subset of the specific example functional modules of encoder 300a shown, additional functional modules, and/or in a different arrangement than illustrated.

FIG. 3(b) is an illustrative diagram of an example next generation video decoder 300b, arranged in accordance with at least some implementations of the present disclosure. FIG. 3(b) presents a similar decoder to that shown in FIG. 2, and similar elements will not be repeated for the sake of brevity. As shown in FIG. 3(b), decoder 300b may include prediction decoder subsystem 330b, filtering decoder subsystem 350b, entropy decoder subsystem 360b, transform decoder subsystem 370b, unpartitioner_2 subsystem 380b, unpartitioner_1 subsystem 351b, filtering decoder subsystem 350b, and/or postrestorer subsystem 390b. Prediction decoder subsystem 330b may include characteristics and motion compensated filtering predictor module 213 and/or intra-directional prediction generation module 214. Filtering decoder subsystem 350b may include deblock filtering module 208, quality restoration filtering module 209, characteristics and motion compensated filtering predictor module 213, and/or prediction fusion filtering module 216. Entropy decoder subsystem 360b may include adaptive entropy decoder module 202. Transform decoder subsystem 370b may include adaptive inverse quantize module 203 and/or adaptive inverse transform module 204. Unpartitioner_2 subsystem 380b may include coding partitions assembler 205. Unpartitioner_1 subsystem 351b may include prediction partitions assembler 207. Postrestorer subsystem 790 may include content post restorer module 218 and/or adaptive picture re-organizer 217.

Entropy decoding subsystem 360b of decoder 300b may perform the inverse operation of the entropy encoder subsystem 360a of encoder 300a, i.e., it may decode various data (types of overhead, motion vectors, transform coefficients) encoded by entropy encoder subsystem 360a using a class of techniques loosely referred to as variable length decoding. Specifically, various types of data to be decoded may include “Coded/Not Coded” data, “Modes and Ref Types” data, “Motion Vector” data, “Prediction and Coding Partitions” data, and “Transform Coefficient” data.

Transform decoder subsystem 370b of decoder 300b may perform inverse operation to that of transform encoder subsystem 340a of encoder 300a. Transform decoder subsystem 370b may include two types of components. The first type of example component may support use of the parametric inverse PHT transform of small to medium block sizes, while the other type of example component may support inverse DCT transform for all block sizes. The PHT transform used for a block may depend on analysis of decoded data of the neighboring blocks. Output

bitstream 111 and/or input bitstream 201may carry information about partition/block sizes for PHT transform as well as in which direction of the 2D block to be inverse transformed the PHT may be used (the other direction uses DCT). For blocks coded purely by DCT, the partition/ block sizes information may be also retrieved from output bitstream 111 and/or input bitstream 201and used to apply inverse DCT of appropriate size.

Unpartitioner subsystem 380b of decoder 300b may perform inverse operation to that of partitioner subsystem 320a of encoder 300a and may include two unpartitioning subsystems, coding partitions assembler module 205 that may perform unpartitioning of coded data and prediction partitions assembler module 207 that may perform unpartitioning for prediction. Further if optional adaptive picture organizer module 104 is used at encoder 300a for region segmentation or slices, adaptive picture re-organizer module 217 may be needed at the decoder.

Prediction decoder subsystem 330b of decoder 300b may include characteristics and motion compensated filtering predictor module 213 that may perform prediction of "inter" signal and intra-directional prediction generation module 214 that may perform prediction of "intra" signal. Characteristics and motion compensated filtering predictor module 213 may allow for increasing predictability by first compensating for other sources of differences (such as gain, global motion, registration) or creation of synthesized frames (super resolution, and projection), followed by actual motion compensation.

Filtering decoder subsystem 350b of decoder 300b may perform multiple filtering of the reconstructed pictures based on parameters sent by encoder 300a and may include several subsystems. The first example subsystem, deblock filtering module 208, may deblock and dither to reduce or mask any potential block coding artifacts. The second example subsystem, quality restoration filtering module 209, may perform general quality restoration to reduce the artifacts due to quantization operation in any video coding. The third example subsystem, characteristics and motion compensated filtering predictor module 213, may improve results from motion compensation by using a filter that may adapt to the motion characteristics (motion speed/degree of blurriness) of the content. The fourth example subsystem, prediction fusion filtering module 216, may allow adaptive filtering of the prediction signal (which may reduce spurious artifacts in prediction, often from intra prediction) thereby reducing the prediction error which may need to be coded.

Postrestorer subsystem 390b of decoder 300b is an optional block that may perform further improvement of perceptual quality of decoded video. This processing can be done either in

response to quality improvement parameters sent by encoder 100, or it can be standalone decision made at the postrestorer subsystem 390b. In terms of specific parameters computed at encoder 100 that can be used to improve quality at postrestorer subsystem 390b may be estimation of film grain noise and residual blockiness at encoder 100 (even after deblocking). As regards the film grain noise, if parameters can be computed and sent via output bitstream 111 and/or input bitstream 201 to decoder 200, then these parameters may be used to synthesize the film grain noise. Likewise, for any residual blocking artifacts at encoder 100, if they can be measured and parameters sent via output bitstream 111 and/or bitstream 201, postrestorer subsystem 390b may decode these parameters and may use them to optionally perform additional deblocking prior to display. In addition, encoder 100 also may have access to scene change, spatial complexity, temporal complexity, motion range, and prediction distance information that may help in quality restoration in postrestorer subsystem 390b.

While subsystems 330b through 390b are illustrated as being associated with specific example functional modules of decoder 300b in Fig. 3(b), other implementations of decoder 300b herein may include a different distribution of the functional modules of decoder 300b among subsystems 330b through 390b. The present disclosure is not limited in this regard and, in various examples, implementation of the example subsystems 330b through 390b herein may include the undertaking of only a subset of the specific example functional modules of decoder 300b shown, additional functional modules, and/or in a different arrangement than illustrated.

Referring to FIG. 4, in one example form, in order to provide the method and system disclosed herein, the quantizer adapter and rate control 400 may include an initial qp generator 402 that receives frame data and pyramid rankings for each frame as explained below, and then generates an initial, default Qp for each frame. The complexity of the frame is determined by a complexity value generator 404, and a Qp limit unit 406 determines limits for the Qp values. A sequence start and static scene Qp calculation unit 408 provides the Qp for static or start frames, while a buffer control 412 uses a buffer model 410 to provide a frame Qp depending on buffer status or fullness. Thus, the buffer control 412 has a buffer fullness unit 414 to determine whether a certain fullness level (or operating point) is being met, and if not a delta Qp unit 416 is provided to adjust the Qp for the frame. The Qp to be used for the frame may then be calculated by a sequence non-start/active scene Qp calculation unit 418. By one alternative, in order to provide the one-pass CBR control with lookahead option, the buffer control 412 also may include an optional lookahead temporal adaptive control 420 with a lookahead key frame

boosting unit 422, a lookahead temporal complexity masking unit 424, and a lookahead Qp factor smoothing unit 426, all described in detail below.

Referring to FIG. 5, by another alternative, in order to provide the two-pass VBR rate control, a quantizer adapter and rate control 500 may have additional, or alternative, components compared to the rate control 400, such as a first pass analyzer 502 to determine which frames are the more complex frames, a second pass analyzer 504 to provide a more final frame Qp, and a second pass error correction and bit control 506 to make further adjustments to the Qp. For this configuration, the first pass analyzer 502 may have an initial constant Qp generator 508, a complexity value generator 510, and a Qp limit unit 512. The second pass analyzer 504 may have a temporal adaptive control 514 with a key frame boosting unit 516, a temporal complexity masking unit 518, and a Qp factor smoothing unit 520. The second pass analyzer 504 also may have a target constant Qp generator 522 and a final error bits and target bits generator 524. The operation of these elements are described in greater detail below.

Referring to FIG. 6, a flow diagram illustrating an example process 600 is arranged in accordance with at least some implementations of the present disclosure. Process 600 may include one or more operations, functions or actions as illustrated by one or more operations numbered 602 and 604. Process 600 may form at least part of a next generation video coding process. By way of non-limiting example, process 600 may form at least part of a next generation video encoding process as undertaken by encoder system 100 of Fig. 1, rate control 400 or 500, and/or any other encoder system or subsystems described herein.

Process 600 may include " obtain frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past, future, or both references" 602. I-pictures are spatially coded or "intra" pictures. The pictures on the maximum level may be bi-directional or B-pictures. Otherwise, such bi-directional frames may be functional or F-pictures similar to B-pictures except that F-pictures may use modified references that are modified by a morphing technique or a synthesizing technique. Such modification may include morphing

techniques such as gain compensation, blur/registration compensation, or global/dominant motion compensation. Synthesis techniques include projected interpolation or super-resolution.

Process 600 also may include "determine a quantization parameter for a current frame depending at least on the level of the hierarchy of the current frame" 604. Particularly, and as explained herein, the hierarchy or pyramid may be used to determine an initial or default Qp. By one form, a rank (0) (also referred to as base level) Qp is established, and the Qp for the remaining ranks are based on the rank (0) Qp. This may be provided in a one-pass CBR rate control process that adjusts the Qp for some of the frames depending on a buffer status or fullness. Otherwise, the Qp process may be a two-pass VBR process that performs an initial analysis of a frame sequence to determine which frames are more complex (or active), and before performing a second full analysis pass of the frame sequence to finalize the Qp for some of the frames. When sufficient time for two passes is not available, a compromise one-pass VBR rate control with lookahead process may be used that performs an initial analysis on a select number of future frames before performing a full analysis on the entire sequence. More is explained below.

One-pass CBR Rate Control

Referring to FIG. 7 for more detail, an example one-pass CBR rate control process 700 is arranged in accordance with at least some implementations of the present disclosure. Process 700 may include one or more operations, functions or actions as illustrated by one or more operations 702 to 758 numbered evenly. Process 700 may form at least part of a next generation video coding process. By way of non-limiting example, process 700 may form at least part of a next generation video encoding process as undertaken by encoder system 100 of Fig. 1, rate control 400, and/or any other encoder system or subsystems described herein.

Process 700 may include "perform rate control (RC) startup" 702. Preliminarily, the process may, or this operation 702 may include, initialization of rate control parameters such as the rate control mode to be used constant Qp coding (without rate control to vary the Qp for example), one-pass CBR mode, two-pass VBR mode, or one-pass (lookahead) CBR mode. The parameters also may include a target bitrate (in Kbps) and a maximum buffer size in Kb. Based on these command line parameters, different rate control parameters (buffers, estimation models, etc.) are set to appropriate values. Based on the bitrate and the resolution of the sequence, initial default Qps are derived based on predefined tables that factor in complexity of the frames, and that differ depending on the hierarchy or pyramid rank of the frames.

Referring to FIG. 8, a hierarchy first may be explained with a frame or video sequence 800. Such a frame sequence may form a repeating pattern 802 of eight pictures. Each frame or picture is numbered 1-16, in order of display, or more accurately input video, and not in the order of coding, and labeled with a picture type (I, P, F, or f) where capital F indicates the F-picture may be used as a reference frame, and the small f-picture, at least for this figure, is not used as a reference frame. The superscript on the picture type notation indicates which pyramid or hierarchy level or rank the frame resides, thereby indicating the reference frame dependencies also shown by the dependency arrows 804.

The picture sequence 800 includes a rank zero (0) and a three level pyramid with ranks 1 to 3 (when not including rank (0) in the pyramid level numbers). The base or rank zero (0) frames may include I-picture 0 (or in other words at time 0), which is a reference frame to P-picture 8, which itself is a reference frame to P-picture 16, all of which are rank zero as shown by their superscripts. Each pattern 802 has one (rank 1) reference F-picture 4 or 12 in a first level of the hierarchy, two (rank 2) reference F-pictures ((2 and 6) or (10 and 14)) in a second level of the hierarchy, and four non-reference (rank 3) f-pictures (1, 3, 5, 6 or 9, 11, 13, 15) in a third level of the hierarchy. In this example, in the period levels 1 to 3, each frame has a reference from a level that is one lower in rank value, although it need not always be this way. Dependency from a current frame to its reference frame(s) also may jump levels, or may have additional dependencies on the same level. The method and system disclosed herein uses the concept that maximum compression is better approached without significantly sacrificing image quality by using higher compression as a frame is on a level father from the base or rank (0) level.

With this arrangement in mind, and for better rate control, Qp modulation is used within the pyramid coding structure in the RC startup operation 702 to form default constant Qp for encoding for video coding, such as for an NGV Encoder.

Referring to FIG. 9, a start-up process 900 is shown for deriving the initial Qp values based on the hierarchy ranks for the frames, and may be performed by a picture quantizer unit or initial Qp generator 402, by one example, to perform the RC startup operation 702. Generally, the rank (0) frames are provided a Qp first, and then the other ranks are provided a Qp based on the rank (0) frames. Thus, starting with operation 902 on the rank (0) frames, the Qp, such as that which may b specified at a command line (-q N), is assigned 904 to P-picture luma quantizer (inter_q[0]). An offset is added to derive P-picture chroma quantizer (inter_q[1]) based on a default mapping if not directly specified through a command line. Thus, it will be understood

that the following calculations and processes are mainly concerned with the luma Qp, and an offset not provided in detail herein, may be added to the Qp values to obtain the relevant chroma value. It will be appreciated that the same or similar processes could be applied to chroma values which are then converted to luma values instead. The offset for luma/chroma conversion will no longer be referred to herein.

Skipping for now how the initial default P-picture rank (0) Qp (inter_q[0]) is obtained and which is explained farther below, rank (0) I-picture and F-picture quantizers are derived or determined 904 and 906 from inter_q[0] using predefined mapping tables provided below in pseudo code.

$$\text{inter\_q}[0] = N \quad (1)$$

$$\text{intra}_{q[0]} = \text{AutoIQ}(\text{inter}_{q[0]}) \quad (2)$$

$$\text{fpic\_q}[0] = \text{AutoFQ}(\text{inter\_q}[0]) \quad (3)$$

where N is the assigned default value for inter_q calculated below. Example AutoIQ and AutoFQ mapping tables are shown below to obtain the I-picture (intra) and F-picture (fpic) Qp values where the upper value is the inter value above its corresponding intra or fpic value:

AutoIQ (intra Q) Mapping:

```
const double AutoIQ[128] = {
// 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
1.00,1.00,1.75,2.75,3.50,4.50,5.25,6.25,7.00,8.00,9.00,9.75,10.75,
// 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
11.50,12.50,13.25,14.25,15.00,16.00,17.00,17.75,18.75,19.50,20.50,21.25,
// 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
22.25,23.00,24.00,25.00,25.75,26.75,27.50,28.50,29.25,30.25,31.00,32.00,
// 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48,
33.00,33.75,34.75,35.50,36.50,37.25,38.25,39.00,40.00,41.00,41.75,42.75,
// 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
43.50,44.50,45.25,46.25,47.00,48.00,49.00,49.75,50.75,51.50,52.50,53.25,
// 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72,
54.25,55.00,56.00,57.00,57.75,58.75,59.50,60.50,61.25,62.25,63.00,64.00,
// 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
65.00,65.75,66.75,67.50,68.50,69.25,70.25,71.00,72.00,73.00,73.75,74.75,
// 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96,
75.50,76.50,77.25,78.25,79.00,80.00,81.00,81.75,82.75,83.50,84.50,85.25,
// 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
86.25,87.00,88.00,89.00,89.75,90.75,91.50,92.50,93.25,94.25,95.00,96.00,
// 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120,
```

```
97.00,97.75,98.75,99.50,100.50,101.25,102.25,103.00,104.00,105.00,105.75,106.75,
// 121, 122, 123, 124, 125, 126, 127
107.50,108.50,109.25,110.25,111.00,112.00,113.00
};
```

AutoFQ (fpic Q) Mapping:

```
const double AutoFQ[128] = {
// 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
1.00,1.00,2.00,3.00,4.00,5.25,6.50,7.75,9.00,10.25,11.50,12.75,14.00,
// 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
15.25,16.50,17.75,19.00,20.25,21.50,22.75,24.00,25.25,26.50,27.75,29.00,
// 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
30.25,31.50,32.75,34.00,35.25,36.50,37.75,39.00,40.25,41.50,42.75,44.00,
// 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48,
45.25,46.50,47.75,49.00,50.25,51.50,52.75,54.00,55.25,56.50,57.75,59.00,
// 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
60.25,61.50,62.75,64.00,65.25,66.50,67.75,69.00,70.25,71.50,72.75,74.00,
// 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72,
75.25,76.50,77.75,79.00,80.25,81.50,82.75,84.00,85.25,86.50,87.75,89.00,
// 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84,
90.25,91.50,92.75,94.00,95.25,96.50,97.75,99.00,100.25,101.50,102.75,104.00,
// 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96,
105.25,106.50,107.75,109.00,110.25,111.50,112.75,114.00,115.25,116.50,117.75,119.00,
// 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
120.25,121.5,122.75,124.00,125.25,126.50,127.75,127.75,127.75,127.75,127.75,127.75,
// 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120,
127.75,127.75,127.75,127.75,127.75,127.75,127.75,127.75,127.75,127.75,127.75,127.75,
// 121, 122, 123, 124, 125, 126, 127
127.75,127.75,127.75,127.75,127.75,127.75,127.75
};
```

Once the values for intra_q[0] and fpic_q[0] are obtained from the tables, as mentioned above the chroma quantizers for I and F-pictures are also derived by adding chroma offsets to corresponding luma quantizers.

For Pyramid coding by one example approach, these default Qps are used only for ZERO_RANK frames (rank (0)), *i.e.*, inter_q[0] for zero rank P-picture and intra_q[0] for Zero rank I-picture. Once the first initial default Qp values for rank (0) are set, the process 900 may perform pyramid coding, which starts with determining whether a maximum rank r has been reached 908. If not, by one possible example, the rank r is set to r+1 (operation 910). For any frame of higher rank 'r' (where $r \neq 0$), the corresponding Qp is derived (912) from using (914) the following equation:

$$\text{Qpr} = \text{inter\_q}[0] + ((\text{fpic\_q}[0] - \text{inter\_q}[0]) + \frac{1.0}{2.0}) * (\text{r}) \quad (4)$$

where ((fpic_q[0] – inter_q[0]) + 1.0/2.0) is considered a delta factor. Thus, after rank r = 0, rank r is set to 1 for the use of equation (4) recited above. This equation is good for both P and F pyramid coding, and with this equation then, the frames on the same rank start with the same initial or default Qp value (where each rank has a Qp value). That is the Qp for any frame of higher rank is derived by adding a fixed delta value to the initial default Qp of its immediate previous (or lower) rank reference frame and irrespective of its frame type. Another way to say this is that instead of multiplying the delta factor by r as in equation (4), adding the delta factor to the Qp of the adjacent value of lower rank relative to the current rank will obtain the same result. When the max r value is reached, the process 900 may provide 918 initial/default Qps for quantization adaption control. It will also be noted that the equation is used even when there is not actually any F-picture on the rank (0) or base level.

With hierarchical or pyramid coding, by one example, further adaptive quantization may be performed only on MAX_RANK F-pictures (rank of the highest value in the pyramid). For any other rank, Qp is derived as in equations (1) to (4) above, and is used irrespective of its frame type. Whereas for MAX_RANK F-pictures, frame Qp derived as in equation (4) may be further adapted 916 to a region type based on its classification (flat/mixed/textured) before it is used.

Referring to FIG. 10, an example frame sequence 1000 is provided to demonstrate equation (4). Frame sequence 1000 has a frame sequence similar to that of sequence 800 (FIG. 8) with 16 frames and a hierarchical Qp structure with F-pyramid coding where only F-pictures form the higher levels other than rank (0) level. The lines connecting the frames in order from one to sixteen merely convey the input video order and do not necessarily convey any reference dependency or coding order. Here, it is assumed that the assigned default Qp for inter_q[0] is from a specified command line P Qp is 8 (*i.e.*, -qp 8). This is made up for this example and provides no actual limitation to Qp values. While inter_q[0] = 8. The other derived Qps, may be intra_q[0] = 7, and fpic_q[0] = 9. By this example, I-pictures are of ZERO rank and coded with intra_q of 7. P-pictures are of ZERO rank and coded with inter_q of 8. Inserting the default inter_q[0] and fpic_q[0] into equation (4) above, F-pictures of rank 1 are coded with a derived Qp of 9.5, rank 2 frames are coded with a derived Qp of 11, and rank 3 frames are coded with a derived Qp of 12.5.

Referring to FIG. 11, a hierarchical Qp structure with P-pyramid coding for a specified command line P Qp of 8 (*i.e.*, -qp 8) is provided. For a P-pyramid, P-pictures may occupy levels

in addition to the rank (0) level except for the maximum (MAX_RANK) level 3. For this scenario, the default Qp is assigned to inter_q[0]. So, by one example, inter_q[0] = 8, while the other derived Qps are intra_q[0] = 7, fpic_q[0] = 9. I-pictures are of ZERO rank and coded with intra_q of 7. P-pictures of rank 0 are coded with derived Qp of 8. Rank 1 frames are coded with derived Qp of 9.5. Rank 2 frames are coded with derived Qp of 11. F-pictures of MAX_RANK rank (3) are coded with derived Qp of 12.5.

Maintaining this pyramid structure of Qp variation between ZERO_RANK reference frames (which typically may vary as much 8, 12, 16 or 20 between Qp values of adjacent levels) also preserves the gains of pyramid coding in rate control. This suggests that the rate control should modulate the ZERO_RANK frame Qps according to the buffer status, while the rest of the quantizer parameters will be derived within the pyramid structure as explained above. However, with this pyramid coding structure, the buffer control model should keep the average bit rate close to the target bit rate mainly by regulating the Qps at zero rank frames.

So herein, the rate control may be used to find and maintain a mean ZERO_RANK P Qp which corresponds to the target bit rate. The buffer control modulates this mean P Qp as the buffer fluctuates due to scene complexity and buffer constraints. Also, whenever the mean P Qp is modified, all of the corresponding Qps for different frame types and higher rank frames should be recalculated using the last updated P Qp.

Returning now to the calculation of the initial default rank (0) P-picture Qp (inter_q[0]) or initial Qp estimation (operation 902 of FIG. 9), this operation may also be considered to be part of, or includes, the operation to calculate average temporal complexity 704 for picture rate control, and may be performed by an average complexity generator 404. This is part of frame rate control before encoding of each frame which decides the current frame Qp. While other methods of rate control start with a pre-fixed constant Qp for any sequence irrespective of the target bit rate, this does not always achieve a good quality target bit rate. To achieve a target bit rate with better or the best quality, rate control may be initiated with the correct Qp based on the complexity of the sequence, its resolution, and the target bitrate. Thus, based on all the future input frames that are read into buffers so far in the pdistance module, an average spatio temporal complexity can be computed and used to provide the initial default inter_q[0] as follows.

Based on several experiments, a relationship is derived between the sequence complexity, target bitrate, and Qp. Several sequences, in common intermediate formats (CIF) or YCbCr video coding format to name a few examples, are encoded with fixed Qps of 8, 16, 32 and 56

respectively for levels 0 to 3 in F-pyramid mode. Based on the averaged data of all CIF sequences, the following relations are derived between these quantities.

Referring to FIG. 12, a simple linear relation (linear in log domain) that fits into this model is:

$$\log Q = (0.827460) * \log(C) + 2.3557763 \quad (5)$$

where logQ = log of estimated Qp, C = (avg_temporal_complexity /target_bitrate), target_bitrate is in bps, avg_temporal_complexity is the average of temporal_complexity of all available future frames, calculated as (avgTSC/(width*height)^0.25) where TSC refers to temporal spatial complexity or temporal complexity measure in short.

The complexity of future frames are available from a PDistance module, which may be part of adaptive picture structure organizer 104 by one example, and that reads at least a few frames ahead into the future before deciding the current frame type (Pdistance is the number of frames between P-pictures). Thus, the temporal complexity measure (TSC) is already computed for future frames as a complexity measure. This has an average prediction error of 9.66 for all sequences.

Referring to FIG. 13, a more accurate quadratic relation (in log domain) is

$$\log Q = (0.053992) * \log(C) * \log(C) + (0.979245) * \log(C) + 2.45758 \quad (6)$$

which has an average prediction error of 10.76 for all sequences.

The final Qp is derived as

$$Qp = \text{power}(10, \log Q) \quad (7)$$

This estimated Qp is used as the initial Qp for encoding. This is assigned to ZERO_RANK P-picture Qp, such as inter_q[0] also to referred to herein as Pqp or PQp. From this, the rest of the default initial, and constant, Qps for different frame types and higher ranks are derived from equations (1) to (4).

Process 700 may then continue with calculating 706 Qp limits. The initial Qps derived from the above models are subject to some general limits, which may be determined by a QP limit unit 406 for one example. For the initial few frames, and until some history is built into the

estimation models, the process 700 may include provide 708 QP limits for sequence start such that the limits are derived from predefined (Qp estimation) tables (FIGS. 14-18) which are calculated based on the observation of different sequences. More specifically, for the absolute or very beginning of a sequence, the initial Qp estimation tables were developed and based on experiments used to derive the boundary Qps. These tables are a map of complexity, bitrate (bpp) and ZERO_RANK frame Qp which tries to predict the mean ZERO_RANK P-picture Qp for the complete sequence with the calculated average complexity which can produce the specified target bitrate. Qp max and min limits are then derived from the corresponding max and min predefined tables as explained below.

A set of sequences are encoded with default constant Qp coding, and the relation between the Qp, bitrate produced (in terms of bits per pixel (bpp) and indexed as bpp_id), and the average spatio-temporal complexity (avgTSC) was observed. A table was compiled based on such observations for different sequences which approximately follow this trend.

For calculating the bitrate index bpp_id, based on the specified bitrate, frame rate and resolution of the sequence, bits per pixel can be calculated as:

$$\text{bpp} = \frac{\frac{\text{bitrate}}{\text{frame rate}}}{\text{width*height}} \tag{8}$$

where width*height is the resolution of a frame. Further, the bitrate index (bpp_id) is derived using the table of FIG. 14 “bpp to bpp_id mapping” to reduce exact bpp values to one of nine possible whole numbers 0 to 9. The bpp values shown on the table are the maximum values for that range or index number. Thus, for example, bpp_id is 0 for any value from 0 to 0.0152, and so forth.

Similarly, in order to determine an average spatio temporal complexity measure index (cpx_id), first motion compensation residual energy between successive frames is used to determine a spatio temporal complexity measure (cpx). This measure is computed for several successive frames in the input sequence (up to a maximum pdistance of about twenty frames and a minimum of about five frames), and an average is computed as avgTSC. Then, the complexity index (cpx_id) is derived for the average value (avgTSC) using the table of FIG. 15 “cpx to cpx_id mapping” to determine the range or whole number, here numbered 0 to 9, of cpx_id to which it corresponds. The avgTSC values shown on the table are the maximum values

for that index number, such that cpx_id = 0 covers avgTSC of 0 to 0.75, and so forth, except that anything over a 9.25 (INT_MAX) corresponds to the 9 index number.

Once the index values are obtained for the bitrate and complexity, mean Qp can be determined from "mean QP table" (or MeanQp[cpx_id][bpp_id]) FIG. 16, "minimum QP table" (or MinQp[cpx_id][bpp_id]) FIG. 17, and "maximum QP table" (or MaxQp[cpx_id][bpp_id]) FIG. 18, where the cpx_id values are along the side of each table, and the bpp_id values are along the top of the table.

The minQp and maxQp tables are used to derive the Qp bounds as follows:

$$\text{minQp} = \text{MinQp[cpx\_id][bpp\_id]} \tag{9}$$

$$\text{maxQp} = \text{MaxQp[cpx\_id][bpp\_id]} \tag{10}$$

These limits for limiting the rate control Qp are used on some but not all of the frames because this criterion does not consider the buffer status. So this may result in the actual bit rate deviating from the target bitrate.

The process 700 may otherwise include establish 710 QP limits based on history. After some history is established of past total bits consumed for the average complexity and the average Qp used, Qp for a new frame may be based on the projected complexity and bits, including the current frame, relative to the past statistics, which set the total bit consumption, including the current frame, close to the target rate. The Qp limits are derived from this Qp using a standard P to I and P to F Qp mapping. Thus, bits and Qp models, and minimum and maximum Qps may be derived using the following equation:

$$\text{Qp} = (\text{act\_bits} * \text{avg\_pqp} * \text{cplx})/(\text{target\_bits} * \text{avg\_cplx}) \tag{11}$$

where avg_pqp is the average Pqps of all the frames encoded so far, avg_cplx is the average spatio temporal complexity, act_bits is the total bits consumed for all the frames encoded so far, and when cplx is the complexity of the current frame, target_bits is the total target bits for all the frames including the current frame. This results in a 'Qp' required to encode the current frame which may result in taking the act_bits close to target_bits. Generally, a drift of about 15% of avg_pqp is allowed for Qp for successive frames. If the resulting change in Qp is above this limit, then up to a maximum drift of about 25% of pqp is allowed when this is the

result of a sudden change in complexity (by above about 25% of avg_cplx). This Qp is assigned as mean Qp.

It has been observed that the above criterion may result in abrupt change in mean and boundary Qps when there is a lot of change in either complexity or bit consumption. To overcome this, use of the above criterion may be limited to a special case where the Qp and complexity deviation are within certain limits. In other words, the history alternative should be limited to situations when the current frame complexity and the actual bitrate so far are relatively close to the average complexity so far and the target bitrate respectively.

In a third alternative (which may be considered the default), for the remainder of the sequence is to use 712 the average or default P Qp, so far of encoded frames in the sequence, directly as the mean Qp. Then, whether the default mean Qp here, or the history based mean QP is established as above, the Qp max and min limits are derived from these Qp using the standard P to I and P to F Qp mapping based on AutoIQ and AutoFQ mapping in pseudo code as recited above:

$$\text{minQp} = \text{AutoIQ}[\text{mopQp}] \tag{12}$$

$$\text{maxQp} = \text{AutoFQ}[\text{mopQp}] \tag{13}$$

where mop is mean operating point. Then, the Qp derived from equation (7) is clipped to these limits:

$$\text{Qp}_{\text{clip}} = \text{clip}(\text{Qp}, \text{minQp}, \text{maxQp}) \tag{14}$$

which will set the Qp_{clip} at the closest limit if it is originally beyond the limits.

The process 700 then continues with determine frame QP 714. By one form, this may first include checking a frame to determine whether it is a blank or static scene 716. This may be performed by determining whether a certain number of successive frames have the same pixel data. If the scene is static or a series of blank frames, then the default, fixed Qp is used. As mentioned above, this may include providing 718 the average P Qp, frame type, and rank to determine 720 the Qp, and in one example, by a static/blank scene calculation unit 408. Even if this happens in the middle of a scene (for instance when the scene becomes static/blank for a short time), then the last stable state Qps are used for all frames so far (or up to this point) that are detected as blank/static frames.

Also, it may be determined 722 whether a frame is the beginning of a sequence by order number or detection of a scene change by known motion analysis techniques for example. When a frame is determined to be the beginning of a sequence or scene change, the average complexity, target bitrate (bpp) and developed Qp modeling based on experimentation are provided 724. The more final, or final, frame Qp of a start sequence, may be calculated by a sequence start calculation unit 409 by one example, is the same calculations 726 as for the default constant Qps. Thus, the Pqp for rank (0) may be the inter_qp from equation (7), and the remaining levels or ranks receive their Qp from equation (4) as explained above. This Qp is constrained to the limits as derived in the previous QP limit section (equations (8) to (14)). This may be performed because sometimes the model may predict a wrong Qp due to abnormal complexity in the beginning (blank frames, etc.).

Otherwise, as described below, the buffer status may be determined using bit estimation for frames. Specifically, bit estimation for initial or beginning sequence frames may include the use of pre-computed bit estimation models that are used to estimate the target bits for the initial few frames until linear estimation models (which are used subsequently for bit estimation) get adapted through at least one data point. For I-picture coding, the first I-picture is coded using the Qp derived in equation (11). For P-picture bit estimation, if the current frame is a P-picture (only ZERO rank for F pyramid for example), then the P estimated bits are derived based on the following criterion.

Referring to FIG. 19, a chart labeled “I to P bits ratio vs Qp” is provided. Since a P-picture can be coded only after the first I-picture is coded, the P-picture bits can be estimated based on the I-picture bits. Again, based on several experiments, the following relationships are derived between the ratio of I to P-picture bits (ip_ratio) and the Qp.

Linear relationship:

$$\text{ip\_ratio} = 0.0256 * \text{Pqp} + 2.0735 \tag{15}$$

Quadratic relationship:

$$\text{ip\_ratio} = -0.0007 * \text{Pqp} * \text{Pqp} + 0.0735 * \text{Pqp} + 1.5559 \tag{16}$$

where the ip_ratio can be derived using one of the above relations, and where Pqp is the P-picture Qp value. The estimated bits for P-pictures can be derived using the following relation.

$$P\_est\_bits = I\_act\_bits/ip\_ratio \quad (17)$$

where the above modeling is based on the average of CIF data, and I_act_bits is the actual I-picture bits.

Similarly, for F-picture bit estimation, if the current frame is an F-picture, then the F estimated bits are derived from the future anchor P-picture either as actual bits (if coding is completed) or estimated bits. Here, anchor means the next P-picture after the current F-picture in input video order that is a reference frame. The following linear relationships between p and f bits were determined depending on the non-zero frame rank in the hierarchy or pyramid.

$$pf\_ratio = 0.0404 * Pqp + 1.4593 \text{ (for rank 1)} \quad (18)$$

$$pf\_ratio = 0.0464 * Pqp + 4.4129 \text{ (for rank 2)} \quad (19)$$

$$pf\_ratio = 0.0464 * Pqp + 12.568 \text{ (for rank 3)} \quad (20)$$

where the pf_ratio can be derived using one of the above relations. The estimated bits for P-pictures can be derived using the following relation.

$$F\_est\_bits = Futr\_anchor\_bits/pf\_ratio \quad (21)$$

where Futr_anchor_bits is the actual bits of the anchor P-picture.

Once bit estimators are updated as introduced and explained below, linear bit estimation models may be used for rate control near the beginning of the sequence rather than using the ratio method. More specifically, simple linear estimation models are used in rate control to project the estimated bits of a given frame (of a specific type (P- or F-) and rank) for its given complexity and the current mean Qp. As mentioned in the first section of the algorithm, the Qp varies in a hierarchical structure with pyramid coding. Frames with the same rank and type are wider in distance (along the input video order as in FIGS. 8 and 10-11) such that Qp between successive frames (on different ranks and possibly of different type) may vary widely. This rapid variation of Qp between successive frames, which in turn results in wide variation of bits among successive frames, complicates the linear estimation model estimation and update.

So separate linear estimation models are used for different combinations of frame type and rank. In total, about five valid bit estimators (such as for (picture type)(rank) of I0, P0, F1, F2 &

F3) exist in F pyramid coding. These linear estimators are not used until they are updated with at least one data point to derive the initial parameters.

Each estimator consists of three parameters: (1) value, the model parameter, (2) weight where weightage is given to the past average compared to the present sample while updating the model parameters, and (3) length which increments with each data point and is used to find the average value of the model parameter.

The estimation of bits may be performed for a given frame (of specific type & rank) with spatio temporal complexity, Cplx for qp in the equation (for example, the bit estimator):

$$Est\_Bits = (Cplx * Value) / (length * qp) \quad (22)$$

The updating of the bit estimators includes the following. After actual coding of each frame, if actual bits produced are bits for given Qp, the corresponding estimator model parameters are updated as follows

$$Value \mathrel{*}= weight \quad (23)$$

$$Length \mathrel{*}= weight \quad (24)$$

$$Value \mathrel{+}= (bits * qp)/Cplx \quad 25)$$

$$Length++ \quad (26)$$

where the initial values for model parameters include Length = 0, Value = 0, and Weight = 0.25 for I-pictures, and 0.40 for other picture types. The 'TSC' measure computed in Pdistance module is used as a complexity measure of a frame Cplx. It should be noted that a number of the equations herein are recited in programming language such that X += A means assignment of X = X + A, X *= A means X = X * A, X -= A means X = X – A, and so forth, and "++" means to increment such as by 1).

When the current frame is not a beginning sequence frame or static/blank frame, the process 700 turns to a buffer control 412 for example, to provide 728 frame Qp based on buffer status. Thus, for any other frame, buffer control determines the frame Qp. By default the current mean operating Qp should be used for coding the current frame if the current buffer status is close to the operating point, and the complexity of the current frame is similar to the average

complexity so far. Otherwise, a delta qp value is derived that may be applied to the current mean Qp which will take the buffer close to the mean operating point. This is derived as follows.

Referring to FIG. 20, let Pqp be the target ZERO_RANK P-picture quantizer parameter that can give the ideal target bit rate. Before encoding of each frame, buffer control may use a buffer model 410 and a buffer fullness unit 414, by one example, to project the buffer status if the current frame is encoded with the Qp derived from the current Pqp using Equation (4) and depending on its type and rank. Depending on the estimated buffer status, the current Pqp is either left unmodified or modified so as to avoid buffer overflow or underflow. The buffer control tries to modulate the current Pqp to keep it close to the target Pqp, whenever there is buffer fluctuation due to variation in scene complexity. So, keeping the buffer at a mean operating point (buf_mop) is equivalent to maintaining the Pqp close to a mean Qp operating point for a scene of given complexity. Whenever Pqp is modified by buffer control, all the Qps corresponding to different frame types and ranks are updated using Equation (4) above.

To accomplish this, the buffer control may first include initialize buffer 730 where the different parameters corresponding to buffer control are initialized to appropriate values as follows, and for which are shown on the buffer or buffer model 2000 (FIG. 20) where the vertical position on the page shows the relevant fullness levels:

$$\text{Upper limit of buffer: } \text{buf\_max} = \big((\text{bufferSize})?\,(\text{bufferSize}):(2 * \text{bitrate})\big) \quad (27)$$

$$\text{Lower limit of buffer: } \text{buf\_min} = (0.00 * \text{buf\_max}) \quad (28)$$

$$\text{Mean Operating Point of buffer: } \text{buf\_mop} = (0.40 * \text{buf\_max}) \quad (29)$$

$$\text{Current buffer fullness: } \text{buf\_ful} = \text{buf\_mop} \quad (30)$$

$$\text{Average Buffer fullness: } \text{avg\_buf\_ful} = (0.50 * \text{buf\_max}) \quad (31)$$

$$\text{Target bits per frame: } \text{target\_rate} = (\text{bitrate}/\text{framerate}) \quad (32)$$

where buf_ful is current buffer fullness. The process 700 may continue with estimate 732 buffer. Consider a frame sequence such as a sub-group of pictures (SGOP), also referred to as a coding unit, and located between two ZERO_RANK reference frames. Another way to say this is that the SGOP may be a set or sequence of pictures or frames from the future ZERO_RANK frame to the frame succeeding the past ZERO_RANK frame. With the current state of buffers, Qps, and estimation models, the buffer status may be projected at the end of the SGOP.

For all the frames in the current SGOP, bits are estimated (fr_est_bits) using either of the bit estimation methods used for initial (sequence beginning/start) frames. Specifically, it is determined whether a history exists 734 such that bit estimators as described above are updated with at least one data point to derive initial parameters. If an insufficient history exists, bit ratios 736 are used which includes some proportion models derived from experiments to estimate the frame bits with the given complexity and Qp. These bit proportion models include I to P bit ratios, P to F bit ratios, and so forth as explained above with equations 15 to 21.

When a sufficient history is built into the bit estimation models, these can be used to estimate the bits a frame is going to consume for its complexity. Specifically, in this case, the linear estimation model is used 738 (equations 22 to 26), for example, and depending on the current state of the coding. Then, the current Qp corresponding to the frame type and rank may be used on the frame. The Qp is derived from the current Pqp. For those frames which are already coded in the current unit, their actual bits are already updated in the buffer.

Once a bit prediction is established 740, buffer increments, levels, and threshold can be calculated 742. The current buffer and estimated buffer levels are mapped to indices on a scale of about, and in one form exactly, -10 to +10, based on whether the current buffer status is below or above the mean operating point.

The threshold of the buffer increment is the maximum increment for a frame instance that the buffer can tolerate at its current level of fullness. The rate of buffer increment at the current frame instance compared to the average buffer level can be estimated. Depending on the current buffer level, a predefined table (FIG 21) gives the threshold to this buffer increment. For all the frames in the current SGOP to be coded, the buffer increment is calculated if these frames were to be coded using the current operating set of Qps. The buffer increment is calculated as:

$$buf\_incr = 0; \tag{33}$$

and for all frames in the SGOP:

$$buf\_incr \mathrel{+}= (fr\_est\_bits - target\_rate) \tag{34}$$

where fr_est_bits is the estimated bits for the current frame using the initial frame bit estimation equations mentioned above, with its temporal spatial complexity and the qp corresponding to the current frame type and rank derived from the current Pqp using Equation (4).

If the current buffer fullness is buf_ful, then the projected buffer status, considering the same Pqp, is:

$$\text{buf\_est} = (\text{buf\_ful} + \text{buf\_incr}) \tag{35}$$

With pyramid coding due to the inherent Qp structure corresponding to different ranks, the frame bits also vary in the same pattern. So there will be wider variation in the estimated buffer if we calculate buffer increment for each frame individually. Instead the idea is to nullify this wide variation by calculating the buffer increment for the entire SGOP, and then modulating the Pqp at each frame based on the estimated buffer status. So the buffer increment is always considered with regard to the buffer status before starting the coding of the current SGOP.

$$\text{buf\_incr} = (\text{buf\_est} - \text{last\_buf\_ful}) \tag{36}$$

where last_buf_ful is the buffer status (buf_ful) at the beginning of the current SGOP. The buffer status is updated at the instance of each ZERO_RANK reference frame and is considered the buf_incr from this point on for each frame in the SGOP.

Once the bit prediction is obtained for the buffer, and the buffer levels, increments, and thresholds are established, process 700 may continue with determine 744 delta Qp, performed by, in one example, a delta Qp unit 416, in order to set a maximum adjustment of the Qp. In more detail, a mean operating point is selected in the buffer initially. For example in Eq. (30), buf_ful is set to buf_mop which is about 40% level of the total buffer (buf_max). Whenever the current buffer fullness (buf_ful) deviates from this operating point (buf_mop), an effort is made to bring buf_ful back to the operating point buf_mop, but slowly. It is assumed that the mean operating point of the buffer (buf_mop) corresponds to a mean operating point for Pqp, such that a Pqp_mop exists. So one objective of the buffer control is to maintain the current Pqp close to this mean operating point Pqp_mop, and whenever the current Pqp deviates from Pqp_mop due to a change in scene complexity, the current Pqp may be modulated in small amounts to reach the operating point Pqp_mop.

However, bigger quantizer steps between neighboring frames will cause visual changes in quality. Thus, Pqp is adjusted to match the target operating point in small delta values frame to frame. If buf_ful is the current buffer fullness, buf_est is the estimated buffer level and buf_incr is the increment in the buffer from the beginning of the SGOP, then the delta Qp can be derived as follows.

$$Pqp' = Pqp + delta\_qp \quad (37)$$

where

$$delta\_qp = qp * (delta\_qpf - 1.0) \quad (38)$$

and where

$$delta\_qpf = (1.0 + vbv\_delta) * (1.0 + cplx\_delta) \quad (39)$$

where two factors of delta Qp are derived, one corresponding to the current state of the buffer (vbv_delta, where vbv refers to video buffering verifier), and the other one corresponding to the current scene complexity (cplx_delta). These two factors are handled separately, and may respectively be calculated by a vbv unit 420 and cmplx unit 422, because delta Qp may assume different values based on the combination of these two parameters. For vbv_delta, first the state of the current buffer 746 is determined to find whether the buffer increment exceeds a threshold of the current buffer level, and if this threshold is exceeded by one example, then vbv_delta is calculated and used 748. For equation (39) then, the first factor of delta_qpf is vbv_delta corresponding to the current buffer fullness buf_ful, and where the rate of buffer increment, incr_rate is computed as:

$$incr\_rate = (buf\_ful - avg\_buf\_ful)/om \quad (40)$$

where avg_buf_ful is the moving average of buf_ful, and om is the operating margin depending on whether the buf_ful is above or below the buf_mop. Depending on the buffer status a threshold is calculated for a maximum rate of buffer increment. The threshold table (FIG. 21) is used to determine the threshold. Thus, a threshold for incr_rate, brT is as shown on the threshold table (FIG. 21).

If 'om' is the operating margin computed as:

$$om = (buf\_ful > buf\_mop)\ ?\ (buf\_max - buf\_mop) : (buf\_mop) \quad (41)$$

the buffer fullness level may be computed as follows:

$$buf\_full = round((buf\_ful - buf\_mop)*10.0/om) \quad (42)$$

and then the boundary may be checked to determine whether buf_ful is within the boundary:

$$buf\_full = clip(buf\_full, -10, 10) \quad (43)$$

The threshold for the rate of increment of the buffer is computed as

$$rate\_thr = (buf\_ful >= buf\_mop)\ ?\ brT[buf\_full] : brT[(buf\_full * (-1))] \quad (44)$$

Whenever the incr_rate exceeds this threshold for the current buffer level, vbv_delta is applied 748. In that case, the delta Qp weight factor due to buffer status is applied. The weight factor is derived again based on the current buffer level. There are two possible cases for vbv_delta.

In the first case, (buf_ful >= buf_mop), VBV is over consumed, where:

$$vbv\_delta = CLIP(((buf\_ful - buf\_mop))/(4.0 * (buf\_max - buf\_mop)),0.0,0.10) \quad (45)$$

which is applied only if buf_incr > 0.

In the second case, (buf_ful < buf_mop), VBV is under consumed, where:

$$vbv\_delta = CLIP(((buf\_ful - buf\_mop))/(4.0 * buf\_mop), -0.10) \quad (46)$$

which is applied only if buf_incr < 0.

The second factor is the complexity delta (Cplx_delta) that corresponds to buffer increment (or buf_incr), and is the delta Qp weight factor due to the current scene complexity. This has a smaller weightage compared to VBV delta, and is derived to contain the future complexity. Cplx_delta is always applied. Once the buffer is estimated 750 to obtain the buffer parameters (buf_ful, buf_mop, buf_incr, buf_max, and buf_est), there are four possible cases to calculate 752 the complexity delta.

(1) (Buf_ful >= buf_mop) && (buf_incr>=0), VBV is over consumed, and scene complexity asks for more bits:

$$cplx\_delta = CLIP((buf\_incr)/(2.0 * (buf\_max - buf\_est)),0.0,0.05 \quad (47)$$

(2) (Buf_ful >= buf_mop) && (buf_incr<0), VBV is over consumed, and scene complexity is giving back bits to VBV.

$$cplx\_delta = CLIP(((buf\_incr))/(4.0 * (buf\_max - buf\_mop)), -0.025,0.0) \quad (48)$$

(3) (Buf_ful < buf_mop) && (buf_incr<=0), VBV is under consumed, and the frame is depleting the VBV.

$$\text{cplx\_delta} = \text{CLIP}((\text{buf\_incr})/(2.0 * \text{p}{->}\ \text{buf\_est}), -0.05, 0.0) \quad (49)$$

(4) (Buf_ful < buf_mop) && (buf_incr>0), VBV is under consumed, and the frame is filling VBV

$$\text{cplx\_delta} = \text{CLIP}(((\text{buf\_incr}))/(4.0 * \text{buf\_mop}), 0.0, 0.025) \quad (50)$$

Process 700 may then include calculate 754 delta Qp. Both the above factors are combined into a single delta Qp value, and by one approach, in proportion to the current mean Qp.

The process 700 may then include calculate 756 frame Qp, and this may be performed by a sequence non-start/active scene Qp calculation unit 418 in one example. The frame Qp is derived by adding the above delta Qp to the current mean Qp.

The process 700 may also include Qp boundary check 758, where the modified Pqp is subjected to Qp limits as described above with operation 706 to avoid too much deviation between successive frames:

$$\text{Pqp}' = \text{clip}(\text{Pqp}', \text{min\_qp}, \text{max\_qp}) \quad (51)$$

The Qp is then available for frame encoding.

Updating 760 then may be performed. The Qp may be updated whenever P Qp is modified at a frame, such that the Qps corresponding to different ranks are updated using Eq. (4). The Qp corresponding to the current frame type and rank is assigned as est_qp for the current frame which is used for the actual coding of the frame.

A rate control (RC) update may be performed after encoding of each frame is complete, so that the coding state and the buffer status are updated. If the current frame of specific type and rank with complexity cplx is encoded using act_qp, and it produces act_bits, then the buffer and estimation models parameters are updated. For the buffer update,

$$\text{Buffer Update: buf\_ful} \mathrel{+}= (\text{act\_bits} - \text{target\_rate}) \quad (52)$$

For the estimation models update, the linear estimation model that uses bit estimators corresponding to the current frame type and rank is updated as with the sequence start estimation models including equations (23) – (26).

An end of field (EOF) check is performed 762, and if it is not the end of the field signifying the end of a GOP, frame sequence, video, or other predetermined length, then the process is performed again starting with the TSC calculation 706. If the EOF is reached, an RC shut down 764 is performed that clears the rate control parameters.

Modified Buffer Control

An alternate method of buffer control is developed to precisely control the delta_qp for each frame. Here, one of the main ideas is to directly use the estimated buffer status to regulate the frame Qp, instead of using the buffer increment, as well as a few other additions to this algorithm. This modification replaces or modifies the algorithm for determination of the delta Qp and the calculation of the frame Qp thereafter.

As the buffer status is updated after each frame, for buffer control at any frame, this modification applies to the frames that are not yet coded in the current SGOP. Thus, using the current Pqp and the estimation models (for different frame types and ranks), the total buf_incr is estimated for all the frames not yet coded in the current SGOP. The buffer increment is calculated as follows.

$$\text{buf\_incr} = 0; \tag{53}$$

to initially set the buffer increment, and then for all frames in the SGOP,

$$\text{buf\_incr} \mathrel{+}= (\text{fr\_est\_bits} - \text{target\_rate}) \tag{54}$$

where fr_est_bits is the estimated bits for the current frame using the bit ratio models or linear estimation models with bit estimators as for the initial frames, with temporal spatial complexity and the Qp corresponding to the current frame type and rank derived from the current Pqp using Eq (4). If the current buffer fullness is buf_ful, then the projected buffer status (buf_est), continuing with the same Pqp, is:

$$\text{buf\_est} = (\text{buf\_ful} + \text{buf\_incr}) \tag{55}$$

Similar to equations (37) to (40), buffer estimation level (buf_estl) is determined based on the buffer estimation (buf_est) rate threshold for the current level (buf_estl) from the same table (FIG. 21). Similar to equations (42) and (48), vbv_delta is computed as follows.

If vbv_delta is to be applied, there are two cases (buf_est >= buf_mop), where VBV is going to be over consumed, or (buf_est < buf_mop), where VBV is going to be under consumed. In the first case, where VBV is going to be over consumed, two sub-categories exist.

In the first sub-category, incr_rate < (-rate_thr), where the buffer is decrementing rapidly:

$$vbv\_delta = clip((buf\_est-avg\_buf\_est)/((buf\_wt/4.0)*(buf\_max-buf\_mop)), -0.05,0.0) \quad (56)$$

Otherwise, in the second sub-category, the buffer is overflowing:

$$vbv\_delta = clip(((buf\_est-buf\_mop))/(buf\_wt*(buf\_max-buf\_mop)), 0.0,0.10) \quad (57)$$

However, in the second case when (buf_est < buf_mop), and VBV is going to be under consumed, two sub-categories are possible here as well. In the first sub-category, (incr_rate > (rate_thr), and the buffer is incrementing rapidly:

$$vbv\_delta = clip((buf\_est-avg\_buf\_est)/((buf\_wt/4.0)*(buf\_mop)), -0.05,0.0) \quad (58)$$

Otherwise, in the second sub-category here as well, the buffer is overflowing:

$$vbv\_delta = clip(((buf\_est-buf\_mop))/(buf\_wt*(buf\_mop)), 0.0,0.10) \quad (59)$$

where the terms here are defined above except for the following. The final frame Qp is derived the same as in equations (34) to (36).

The main rate control parameters for this modified buffer control include (1) when to apply vbv_delta and buf_wt. As to the first one, the quantizer variation may be well controlled appropriately based on the buffer status and the current rate of variation of the buffer. As long as buf_estl is close to zero not much regulation is needed. As it gets away from this operating point due to variation in scene complexity, a control parameter may be used to control how often vbv_delta is to be applied. Several different criterion may be used based on the buf_estl, frame type, and rank.

For the second parameter, Buf_wt, determines the magnitude of delta_qp. Specifically, Buf_wt controls the quantizer adaptation capability. Again, based on buf_estl, sometimes the quantizer may have to increase or decrease rapidly, and vice versa some other times, depending on the complexity. This parameter decides the magnitude of delta Qps applied for different frames. As mentioned for vbv_delta, this parameter can also be modeled using different criterion based on buf_estl, frame type and rank.

By another approach, this model can be efficiently used with multithreading. Only the rate control and update calls need to be placed using critical sections to allow proper update of rate control buffers and statistics. Since the coding order and lag is random with multithreading, the performance may vary slightly from the single thread case.

One-Pass CBR Algorithm Summary

This section describes the algorithm flow in brief and generally follows FIG. 7 described above.

1. RC Init: Initialize the RC parameters.

2. RC Start Up: Based on the command line parameters, set different rate control parameters (buffers, estimation models, etc.) to appropriate values. Based on the bitrate and the resolution of the sequence, initial Qps are derived based on a predefined table.

3. Pic RC: This is the frame rate control before encoding of each frame which decides the current frame Qp.

A. Avg Complexity: Based on all the future input frames that are read in to buffers so far in the pdistance module, compute average spatio temporal complexity.

B. Check for Static/Blank scenes: If the scene is static or a series of blank frames, then the default fixed Qp is used. Even if this happens in the middle of a scene (for instance when the scene becomes static/blank for a short time), then the last stable state Qps are used for all frames so far they are detected as blank/static frames.

C. Qp limits for the frame:

i. Beginning of a sequence: If it's the absolute beginning of a sequence, then the initial Qp estimation table developed based on experiments is used to derive the boundary Qps. This

table is a map of complexity, bitrate (bpp) and ZERO_RANK frame Qp which tries to predict the mean ZERO_RANK P-picture Qp for the complete sequence with the calculated average complexity which can produce the specified target bitrate. Qp limits are also derived from corresponding predefined tables.

ii. After some initial history is built: Since we have some history of past total bits consumed for the average complexity and the average Qp used, we try to predict the Qp for the new frame based on the projected complexity and bits including the current frame relative to the past statistics which can take the total bit consumption including the current frame close to the target rate and the Qp limits are derived from this Qp using the standard P to I and P to F Qp mapping.

iii. Default: The case above (ii) is only used when the current frame complexity and the actual bitrate so far are relatively close to the average complexity so far and the target bitrate respectively. In any other case, the default Qp is going to be the average Qp so far and the Qp limits are derived from this Qp using the standard P to I and P to F Qp mapping.

D. Frame Qp:

i. Beginning of a sequence or Scene Change: The Frame Qp is derived based on the average complexity, target bitrate (bpp) and Qp modeling developed based on lot of experiments. This Qp is constrained to the limits derived in the previous section C. This is necessary because sometimes the model may predict wrong Qps due to abnormal complexity in the beginning (blank frames etc.).

ii. Buffer Control: For any other frame, buffer control decides the frame Qp. By default the current mean operating Qp should be used for coding the current frame if the current buffer status is close to the operating point and the complexity of the current frame is similar to the average complexity so far. Otherwise, we would like to derive a delta Qp that needs to be applied to the current mean Qp which will take the buffer close to the operating point. This is derived in a series of steps as follows.

a. Buffer Estimation & Buffer Increment: With the current state of buffers, Qps and estimation models, project the buffer status at the end of the current SGOP. An SGOP is a set of pictures from the future ZERO_RANK frame to the frame succeeding the past ZERO_RANK frame. Also for all the frames in current SGOP to be coded, calculate the buffer increment if these frames were to be coded using the current operating set of Qps.

i. History Available: If sufficient history is built in to the bit estimation models, these can be used to estimate the bits a frame is going to consume for its complexity and the current Qp corresponding the frame type and rank is used on the frame.

ii. No History: Otherwise, some proportion models derived from experiments are used to estimate the frame bits with the given complexity and Qp. These bit proportion models include I to P bit ratios, P to F bit ratios etc.

b. Buffer level Indices: The current buffer and estimated buffer levels are mapped to indices on a scale of -10 to +10, based on whether the current buffer status is below or above the mean operating point.

c. Rate of Buffer increment and its threshold: Estimate the rate of buffer increment at the current frame instance compared to the average buffer level. Depending on the current buffer level, a predefined table gives a threshold to this buffer increment, which means it gives the max increment for a frame instance that the buffer can tolerate at its current level of fullness.

d. Delta Qp Calculation: Two factors of delta Qp are derived, one corresponding to the current state of the buffer, the other one corresponding to the current scene complexity. It's necessary to handle these two separately because delta Qp may assume different values based on the combination of these two parameters.

i. VBV Delta: The delta Qp weight factor due to buffer status is applied whenever the rate of increment/decrement exceeds the allowed threshold for the current buffer level. The weight factor is derived again based on the current buffer level.

ii. Complexity Delta: The delta Qp weight factor due to the current scene complexity. This has a smaller weightage compared to VBV delta. This is derived to contain the future complexity.

iii. Delta Qp: Both the above factors are combined in to a single delta Qp value in proportion to the current mean Qp.

e. Frame Qp Calculation: The frame Qp is derived by adding the above delta Qp to the current mean Qp, and limiting to the boundary Qps derived in Section (c).

4. Pic RC Update: This updates the buffer status, estimation models at the end of each frame encode, based on the current frame complexity, Qp and bits.

5. RC Shut down: clears up the rate control parameters.

Two-Pass VBR Rate Control

Referring to FIGS. 22A-22B, a two-pass variable bit rate (VBR) algorithm or process 2200 is shown and arranged in accordance with at least some implementations of the present disclosure. Process 2200 may include one or more operations, functions or actions as illustrated by one or more operations 2202 to 2248 numbered evenly. Process 2200 may form a two-pass rate control mode that may be added to the NGV codec. By way of non-limiting example, process 2200 may form at least part of a next generation video encoding process as undertaken by encoder system 100 of Fig. 1, rate control 500, and/or an additional part of any other encoder system or subsystems described herein.

To implement this process, the first analysis pass, by one example, may be enabled with the command line parameter -bm 3, while the second pass with rate control with the command line parameter -bm 2. A RC startup process may be performed 2202 as with start up process 702.

Now in detail, the two-pass VBR process permits a fixed channel buffer used for a constant bit rate to act as a large buffer with a variable bit rate. This involves two passes, and is accomplished by performing a full first pass analysis 2204 on the complete or entire, or substantially entire, frame sequence to be coded, and as fast as possible. Next, a second pass analysis is performed on the complete frame sequence. The complete sequence is encoded in the first pass in default constant quantizer (CQ) mode. Specifically, the first analysis pass includes determining an inter Qp based on average complexity, target bitrate (bpp) and Qp modeling determined by operation 2206. In other words, by one form, the initial Qp for the first pass is derived in the same way as in the one-pass CBR process to obtain the initial default Qp inter-q[0] using predefined tables (FIGS. 12-13) and the log equation (7) recited above. This may include determining the QP limits as well, and as described above. The default Qps for each of the ranks in the hierarchical structure for the frame sequence then can be calculated by equation (4) as recited above, and these constant values per rank are then considered the resulting Qps for the first pass analysis.

Since it is important to reduce the time duration of the first pass, the first pass analysis may include reducing the size of the video or frames for analysis, and in one form, uniformly reducing frames, to reduce the time for the first pass. This may be accomplished by reducing the

frame or video size to a proportion of the frame's full size, such as one quarter of the full size. In this case, the number of bits of a frame to be analyzed may be reduced by reducing the sampling size so that only one pixel of every 2 x 2 group of four pixels are analyzed for example. The results may then be applied to the full size frame in the second pass (or scaled frame as explained below).

Several frame level statistics, or meta data, like constant frame Qp, bitrates, complexity, and so forth, are written 2208 to a stats (or meta data) file. With this Qp and meta data, the system then knows ahead of time which frames are busy (active versus static), and which frames are complex or problematic. By one example form, the rate controller 500 may have the first pass analyzer 502 with components 508, 510, and 512 to perform these corresponding first pass analysis computations. Thus, by one form, the first pass analysis may perform encoding with the initial default Qps that is sufficient, and in some forms just sufficient, for determining the complexity and bits per frame (or in other words, the bpp and/or bitrate) to determine which frames are more complex (or use more bits) than other frames in the sequence. Thus, for a first pass analysis full encoding for good quality images is not necessarily required, and enhancements for good or high quality encoding may be skipped.

A second pass analysis 2210, in one example, may be performed by a second pass analyzer 504 to form the frame Qps to be used for encoding. To begin, the different frame level stats written into the stats file in the first analysis pass are read 2212 into frame buffers. Before starting the second pass analysis computations, rate control analysis is performed for all of the frames in the sequence to derive the bit target for each frame that would converge the overall bitrate to the target. The algorithm may be used to determine a constant Qp for the second pass which would lead to a target bitrate based on the first pass stats. Adaptive quantization is applied over frame Qps using delta Qp factors derived from key frame boosting and (temporal) complexity masking.

In order to improve the coding efficiency in the second pass, the bits of each frame are scaled 2213 up or down to produce total bits that are the same as the target number of bits per frame. This avoids the need for more time consuming and complicated conversion computations during the encoding. A scale-factor is derived based on the target bitrate and the bitrate produced in the first pass analysis. This scale factor is applied uniformly on each frame to derive the bit target of the frame. Relative bit distribution doesn't change but the overall bit distribution is either scaled down/up by this step.

$$\text{scale\_factor} = \frac{(\text{target\_total\_bits} - \text{total\_ns\_bits})}{(\text{1pass\_total\_bits} - \text{total\_ns\_bits})} \quad (60)$$

where total_ns_bits is the sum of non-scalable bits (headers, etc.) of each frame, and where the target_total_bits is the target bits for a frame including both scalable and non-scalable bits, and 1pass_total_bits is the current actual bit total of the frame derived in first pass analysis. For every frame in the sequence, scaled frame size is derived using the following:

$$\text{scaled\_size} = \text{nsbits} + (\text{frmbits} - \text{nsbits}) * \text{scale\_factor} \quad (61)$$

where nsbits are non-scalable bits and frmbits are the scalabale (image) bits of a frame.

In order to modify the bit target of each frame to improve the quality of encoding in the second pass, a temporal adaptive control 514 may be used to perform adaptive quantization 2214 that may include units 516, 518, and 520 to respectively perform key frame boosting, temporal complexity masking, and Qp factor smoothing explained as follows.

Key Frame boosting 2216 may be performed to raise the quality of static scenes and/or scenes with low complexity without changing the quantizer for the low complexity frames. This is accomplished by providing certain key I-pictures with a Qp that is lower than usual (thereby preserving more detail and better quality on the I-picture). In low complexity scenes, it is observed that if the system preserves the quality of I-pictures by spending extra bits, this will save bits of other frames while still raising the quality of those other frames even though they are coarsely quantized. This is due to a high selection of skip blocks directly transferring the quality of I-pictures across the rest of the frames.

For selection of the scene for boosting, the scene complexity is detected based on the bit proportions and a Qp weighting factor is derived for the I-picture in each GOP.

The bit target of each frame is recalculated after including the additional bits given for key frame boosting in low complexity scenes as follows.

For each frame j following an I-picture k, if fs[j].bits < (fs[k].bits/2),

$$\text{key\_scale\_factor} \mathrel{+}= 0.005 * (j - k) \quad (62)$$

If fs[j].bits < (fs[k].bits/3),

$$\text{key\_scale\_factor} \mathrel{+}= 0.005 * (j - k) \quad (63)$$

where fs[j].bits and fs[k].bits means frame size in number of bits for that frame, and where key_scale_factor is initialized to 1.0. Extra bits for a key frame are derived using the following equation:

$$\text{key\_extra} = (\text{scaled\_size} - \text{nsbits}) * \text{key\_scale\_factor} \quad (64)$$

The scaled_bits for each frame is recalculated to account for these extra key frame bits:

$$\text{scale\_factor} = \frac{(\text{target\_total\_bits} - \text{total\_ns\_bits})}{(\text{scaled\_total\_bits} + \text{key\_extra\_total\_bits} - \text{total\_ns\_bits})} \quad (65)$$

where scaled_total_bits are the total scalable bits for the frame sequence.

Again for each frame, scaled_size is recalculated:

$$\text{scaled\_size} = \text{nsbits} + (\text{scaled\_size} - \text{nsbits}) * \text{scale\_factor} \quad (66)$$

Temporal complexity masking 2218 uses bit shuffling to shift bits from high complexity scenes to low complexity scenes. According to psycho-visual masking concepts, more quality is gained per bit spent in low complex areas relative to high complex areas. So shuffling the bits from high complex areas to low complex areas will increase the overall quality of encoding. This may involve temporal weighting factors, by one example for all of the remaining non-key boosted frames, and that may be derived using relative proportions of bits compared to an average (for respective type and rank).

More specifically, for each frame type and rank combination (such as I0, P0, F1, F2, F3 for a F-picture pyramid with 0 + 3 levels and I, P, and F pictures) in a frame sequence or scene, respective average sizes and relative proportions of bits compared to total bits is computed. For each frame in the sequence of a specific type and rank,

if scaled_size < scaled_avg

$$\text{scaled\_size} \mathrel{+}= (\text{scaled\_avg} - \text{scaled\_size}) * \text{scale\_factor\_spend} \quad (67)$$

if scaled_size > scaled_avg

$$\text{scaled\_size} \mathrel{+}= (\text{scaled\_size} - \text{scaled\_avg}) * \text{scale\_factor\_save} \quad (68)$$

where scaled_size is the bit size of a frame, scaled_avg is the average size for each type-rank in the sequence of the frame being analyzed, scale_factor_spend = 0.1 initially, and

scale_factor save = 0.2 initially. This effectively initially takes 10 or 20% of the difference between the actual size and averaged size of the frame and adds it to the frame size.

The frame target sizes are once again rescaled to target as above after shuffling of bits

$$\text{scale\_factor} = \frac{(\text{target\_total\_bits} - \text{total\_ns\_bits})}{(\text{scaled\_total\_bits} - \text{total\_ns\_bits})} \quad (69)$$

Again, for each frame, scaled_size is recalculated.

$$\text{target\_size} = \text{nsbits} + (\text{scaled\_size} - \text{nsbits}) * \text{scale\_factor} \quad (70)$$

By one example approach, this target size (target_size) is set as the target frame size for each frame in the second pass encoding.

The third technique for improving frame quality, QP factor smoothing, is explained below with a modified version of the two-pass VBR rate control process.

Otherwise, the process may then estimate the frame Qp for each frame. For each frame, rate control determines the frame Qp at the beginning of the encoding of each frame using the following.

$$\text{tgt\_size} = \text{target\_size} - \text{epf} * \text{epc} \quad (71)$$

where epf is error per frame, and in one case epf is 3% of the total bit error (which is the difference between total_act_bits and total_scaled_size updated after encoding of each frame), and epc is error per current frame where the weightage of the current frame type and rank in proportion to the total bits of all frame types and rank.

The new frame qp corresponding to this new tgt_size bits are derived relative to one-pass stats using the following relation.

$$\text{est\_qp} = (1\text{pass\_bits} * 1\text{pass\_qp})/\text{tgt\_size} \quad (72)$$

where est_qp is the estimated Qp for a frame, and is used to encode the current frame, and where 1pass_bits and 1pass_qp are the bits and Qp for each frame from the first pass analysis. After the encoding of each frame, the total bit error is updated which will be used in the next frame Qp calculations

$$\text{total\_bit\_error} += (\text{actual\_size} - \text{scaled\_size}) \quad (73)$$

By another alternative, a modified second pass or rate control pass may be used. The second pass algorithm analysis (versus the first pass analysis) mentioned in the above section could result in wider deviations from target bitrate and peak signal-to-noise (PSNR) ratio losses compared to a constant Qp encoding for complex sequences. More specifically, in the previous algorithm, the first pass bits of each frame are scaled to a target after key frame boosting and (temporal) complexity masking (using bit shuffling). Then, the frame Qp is estimated from these scaled frame sizes. With hierarchical prediction structure in place, this was leading to wide variations of Qps in the second pass, and loss of hierarchical structure in the Qps. Thus, the second pass of the two-pass algorithm is modified as follows to achieve even better performance. The modified algorithm determines a constant Qp for the second pass which would lead to a target bitrate based on the first pass stats. Key frame boosting and (temporal) complexity masking are incorporated using adaptive quantization over the chosen constant Qp. Key frame boosting and frame complexity variations are mapped to a delta Qp factor (qp_deltaf) which can be applied to a frame Qp. The modified algorithm omits the scaling 2213, and is explained in detail in the following sections.

For key frame boosting 2216 in low complex or static scenes for this modified alternative, the key_scale_factor is derived the same or similar as in the previous section (equations (62) – (63)). The determination as to whether a scene is a low complexity scene or sequence is may still be performed by analyzing the proportion of complexity in the sequence, but now this includes the complexities determined for each of the lookahead frames. However, the key_scale_factor now may be mapped to a delta Qp factor as follows.

$$\text{qp\_deltaf[i]} = 1.0/\text{key\_scale\_factor} \tag{74}$$

For temporal complexity masking 2218 for this modified alternative using adaptive quantization, and for each frame type and rank combination, respective average sizes and relative proportions of bits compared to total bits is computed. For each frame(i) in the sequence of a specific type(t) and rank(r),

if fs[i].bits < scaled_avg,

$$\text{qp\_deltaf[i]} = \text{MAX}\left(0.85, \text{pow}\left(\frac{\text{fs[i].bits}}{\text{scaled}_{\text{avg}}}, \frac{1.0}{\text{MAX}(3.0,r+3)}\right)\right) \tag{75}$$

If fs[i].bits > scaled_avg,

$$\text{qp\_deltaf[i]} = \text{MAX}(1.20, \text{pow}\left(\frac{\text{fs[i].bits}}{\text{scaled}_{\text{avg}}}, \frac{1.0}{\text{MAX}(2.0,\text{r}+2)}\right)) \quad (76)$$

where fs[i].bits are the frame scalable bits, and the scaled_avg are the average scalable bits across certain type and rank frames in a frame sequence. Here, the temporal complexities are mapped to Qp delta factors, which provides the proportion of frame to average bits, and at a power sensitive to frame rank. The Qp delta factor then can be used as a scaling factor for the frame Qp.

For Qp factor smoothing 2220, Qp factors for successive frames of the same type and rank are smoothed out using a simple Gaussian filter to avoid wide variation in Qp between successive frames (of the same type and rank) which can lead to rapid variations in quality.

For a chosen sigma,

$$\text{filter\_size} = 1.0 + (\text{int})(\text{sigma} * 4.0) \quad (77)$$

where sigma is chosen as 1.5 to use a 3 taps on either side of the current frame. For a given frame, the filter is applied on a series of samples with (filter_size/2) samples on either side.

$$\text{coeff} = \exp(-\text{d} * \text{d}/(\text{sigma} * \text{sigma})) \quad (78)$$

where d is the index of the current sample with regard to the reference sample.

$$\text{q+= qp\_deltaf[d]} * \text{coeff} \quad (79)$$

$$\text{sum+= coeff} \quad (80)$$

where q and sum are initialized to zero for each reference sample.

Final qp_deltaf is derived as

$$\text{qp\_deltaf[i]} = \text{q/sum} \quad (81)$$

The next operation 2222 in the second pass analysis is to find the target constant mean Qp and bit target for each frame. In one form, this operation may be performed by a constant target Qp generator 522. In this operation, the process uses iteration of a set of possible Qp choices to find the best Qp for the second pass which can result in the target bitrate while applying the delta Qp factors derived in the previous sections (equations (74) to (75) and (81) for example). This

iteration is covered by operations 2224 to 2232. This will result in a target Qp and bits for each frame.

By one approach, the set of Qp choices starts with setting 2224 the coarsest Qp possible which is 128, and progresses in binary order (which divides the step in half each iteration) up to the finest (or smallest) possible Qp step size which is 0.25. Thus:

$$\text{for}(\text{step} = 128; \text{step} \geq 0.25; \text{step} *= 0.5)\ QP += step \tag{82}$$

while skipping those steps which give lesser bits than the target_bits.

$$\text{if}(\text{expected\_bits} < \text{target\_bits})\ QP -= step \tag{83}$$

where target_bits are determined from the target bitrate. For each Qp in the iteration, total expected_bits is computed by calculating the expected_frame_bits for each frame using this new Qp, and accumulating them for all the frames in the sequence being analyzed. The expected_bits are calculated as follows:

For each frame, first the scaled_q corresponding to the current frame type and rank is derived as follows

$$scaled\_q[i] = rankQP[t][r] * qp\_deltaf[i] \tag{84}$$

where rankQP[t][r] are the initial default Qps. The expected bits for the current frame using this scaled_q[i] is

$$\text{scaled\_bits} = (1\text{pass\_bits} * 1\text{pass\_qp})/\text{scaled\_q[i]} \tag{85}$$

$$expected\_bits += \text{scaled\_bits} \tag{86}$$

The Qp that gives the closest bits to target_bits is chosen as the best mean Qp for the second pass. The maximum number of iterations (in terms of Qp step size) to find the best Qp is bounded by

$$\max QP\ step\ iteratios = log2[128] + 2 \tag{87}$$

where the 2 refers to two additional steps corresponding to Qstep = 0.5 and Qstep = 0.25.

In more general terms, initially step (Qstep) is set to 256 and Qp is set to 0. The step is then multiplied by 0.5, and set as the new step. Then, the Qp is added to the step and set as the

new Qp (now at value 128) (operation 2224 and equation (82) above). The estimate total bits are calculated 2226 for the selected frame being analyzed (equation (83 to 86) above). It is noted that both pyramid rank and frame type are factored into the determination of estimate total bits from equation (84) as well as the adaptive quantization. Then it is determined whether a step of 0.25 has been reached yet (2228 and equation (87)). If not, the process then checks 2230 to see if estimated (also called expected) bits are less than the target bits (equation (83)). If so, the process immediately returns to the operation 2224 of obtaining the next new step by multiplying the last step value by 0.5, and then adding the new step to Qp, and the process is looped through again. If the estimated bits are greater than the target bits, first the step is subtracted from Qp (2232) before the process loops back to set a new step and Qp. These operations are set up so that the Qp will vary from less than the target bits, to greater than the target bits, and back and forth until the constant target Qp settles near the target bit value. Once the step reaches 0.125 (one step less than 0,25), the loop is discontinued, and the estimated constant mean Qps that are close to the target bits for the frames are passed on for second pass encoding and bit rate control.

The process 200 then proceeds with second pass encoding and bit rate control 2234, and may be performed by a second pass error correction and bit control 506. This operation is frame rate control before encoding of each frame which may include distribution of final error bits, and calculation of target bits and final current frame Qp. While the initial two-pass algorithm for the constant target QP (operation 2222) gives an estimated scaled_q and estimated QP for each frame, the estimated Qp still needs to be corrected for each frame during actual encoding of the second pass to compensate for prediction errors and bitrate deviations. Note that the calculation of the estimated target bits and constant target Qp in operation 2222 may be considered to be part of the second pass encoding and bit rate control 2234 rather than, or in addition to, part of the second pass analysis 2210. The error correction is applied as follows.

A prediction error correction 2236 includes compensation of a deviation of the actual bits sum (actualBits_sum) from an expected bits sum (expectedBits_sum) of all the frames encoded so far and computed during the initial second pass (using expected_frame_bits). The compensation is applied to the current frame Qp as follows. A temporary buffer is initialized,

$$buf = 0.25 * bitrate \tag{88}$$

$$buf *= MAX(\, sqrt\left(1 - \left(\frac{expectedBits\_sum}{total\_expectedBits}\right)\right), 0.25) \tag{89}$$

where total_expectedBits is the sum of expectedBits of all the frames in the sequence and expectedBits_sum is the sum of expectedBits of all the frames encoded so far.

$$tgt\_err = actualBits\_sum - expectedBits\_sum \quad (90)$$

$$est\_q \ = scaled\_q/MIN(MAX\left(\frac{buf-tgt\_err}{buf}, 0.75\right), 1.5) \quad (91)$$

where tgt_err is the bit sum error, est_q is the estimated Qp adjusted for the bit sum error, and where the scaled_q is the Qp for the current frame derived as in equation (84), and actualBits_sum is the sum of the actual bits of all the frames encoded so far. By one approach, this compensation may be applied selectively on frames giving priority according to their rank where frames at or nearer to the base (rank 0) are given higher priority.

A target bitrate error correction 2238 also is performed so that a deviation of the actual bitrate from the target bitrate is compensated for by adjusting the estimated Qp to derive a more final target Qp, and is compensated as follows

$$est\_q \ast= \left(\frac{actualBits\_sum-total\_nsBits}{targetBits\_sum-total\_nsBits}\right) \quad (92)$$

where est_q is the estimated Qp adjusted here for target bitrate error, and where targetBits_sum and total_nsBits are respectively the sum of target bits and non-scalable bits of each frame encoded so far. The target bits of the current frame are computed 2240 using the new estimated target Qp of each frame (of equation 92) using the bit estimation model from the first pass analysis.

$$\text{target\_bits[i]} = (\text{1pass\_bits} * \text{1pass\_qp})/\text{est\_q} \quad (93)$$

The process may then encode the frame and update 2242 rate control frame statistics at the end of each frame. After encoding each frame(i) different cumulative statistics are updated as follows

$$\text{targetBits}_{\text{sum}} \text{+= target}_{\text{bits[i]}} \quad (94)$$

$$\text{actualBits\_sum+= actual\_bits[i]} \quad (95)$$

$$\text{expectedBits}_{\text{sum}} \text{+= scaled}_{\text{bits[i]}} \quad (96)$$

total_nsBits+= ns_bits[i] (97)

An end of field (EOF) check is performed 2244, and if it is not the end of the field signifying the end of a GOP, frame sequence, video, or other predetermined length, that is being analyzed, then the second pass encoding and bit control process 2234 is performed again starting. If the EOF is reached, an RC shut down 2246 is performed that clears the rate control parameters.

Two-pass VBR Algorithm Summary

This section describes the algorithm flow in brief while generally following FIGS. 22A-22B.

1. Analysis Pass: The complete sequence is encoded in the first pass in default CQ mode. The Analysis pass Inter Qp is derived based on the average complexity, target bitrate (bpp) and Qp modeling developed based on lot of experiments. Several frame level statistics like frame Qp, bits, complexity etc. are written to a stats (or meta data) file.

2. Second Pass: The different frame level stats written in to the stats file in the analysis pass are read in to frame buffers. Before starting the second pass, rate control analysis is done for all the frames in the sequence to derive the bit target for each frame that would converge the overall bitrate to the target. The main idea of the algorithm is to find out a constant Qp for the second pass which would lead to target bitrate based on the first pass stats. Adaptive quantization is applied over frame Qp using the delta Qp factors derived from Key frame boosting and (temporal) complexity masking.

A. Key Frame boosting in low complex or static scenes: For key frames in low complex scenes preserving the quality by spending extra bits would save the bits of other frames. The scene complexity is detected based on the bit proportions and a Qp weighting factor is derived for the I-picture in each GOP.

B. Temporal Complexity masking using Bit shuffling: Similarly the temporal weighting factors for rest of all the frames are derived using relative proportions of bits compared to average (for respective type & rank).

C. Qp factors smoothing: The Qp factors for successive frames of the same type and rank are smoothed out using a simple Gaussian filter to avoid wide variation in Qp between successive frames(of same type and rank) which can lead to rapid variations in quality.

D. Finding the mean Qp for the sequence & Bit target for each frame: In this step, we iterate over a set of possible Qp choices to find the best Qp for the second pass which can result in the target bitrate while applying the delta Qp factors derived in the previous sections over corresponding frame Qp and using bit estimation model to derive the bits for each frame. At the end, this will give target Qp and bits for each frame.

3. Pic Rate Control: This is the frame rate control before encoding of each frame which decides the current frame Qp.

A. Estimated Qp: The initial two-pass algorithm (operation 2222) gives an estimated Qp for each frame. However it needs to be corrected for each frame during actual encoding of the second pass to compensate for prediction errors and bitrate deviations.

B. Prediction Error: The deviation between the actual bits sum and expected bits sum of all the frames encoded so far is compensated on the frame Qp.

C. Bit Target Error: The deviation of the actual bitrate from the target bitrate is also compensated on the estimated Qp to derive the target Qp.

D. Target Bits: This target bits for the current frame are computed based on the target Qp using bit estimation model.

4. Pic RC Update: This updates the cumulative frame statistics at the end of each frame encode, based on the current frame Qp, expected bits, target bits and actual bits.

5. RC Shut down: clears up the rate control parameters.

One-pass CBR Rate Control with Lookahead

Referring to FIG. 23, a one-pass CBR rate control with lookahead process 2300 is shown and arranged in accordance with at least some implementations of the present disclosure. Process 2300 may include one or more operations, functions or actions as illustrated by one or more operations 2302 to 2364 numbered evenly. Process 2300 may form a one-pass lookahead rate control mode that may be added to the NGV codec. By way of non-limiting example, process

2300 may form at least part of a next generation video encoding process as undertaken by encoder system 100 of Fig. 1, rate control 400, and/or an additional part of any other encoder system or subsystems described herein. The descriptions of many of the operations of the one-pass CBR rate control with lookahead are already covered or included by the one-pass process 700 (FIG. 7) in one form or another, and in these cases are not repeated here. The same or similar operations and/or components in both processes are numbered similarly (for example, perform RC startup is operation 702 for process 700 and 2302 for process 2300).

While two-pass rate control provides the best quality, it also causes a significant delay due to analysis of the full frame sequence twice to form the first and second pass analyses. Thus, this alternative one-pass CBR lookahead rate control mode is proposed as a compromise between the one-pass CBR and two-pass VBR in terms of quality, duration, and complexity. This mode provides partial lookahead into scene complexity and preliminary motion analysis in terms of future frames relative to a current frame in the same sequence being analyzed. This accomplished by using lookahead buffers to better control the quality and bitrate. The overall rate control algorithm is similar to the algorithm mentioned for one-pass CBR rate control (process 700) with modifications as outlined in the following sections.

Now in more detail, process 2300 proceeds similarly for many of the operations including perform rate control startup 2302 which may include initialize the RC parameters and, based on the command line parameters, set different rate control parameters (buffers, estimation models, etc.) to appropriate values. Based on the bitrate and the resolution of the sequence, initial default Qps are derived based on a predefined table, and equations (7) and (4) as with process 700.

However, the startup operation 2302 also includes pre-analyzing 2303 lookahead frames. Preliminarily for lookhead, a new command line parameter '-la' is added to specify the lookahead latency or buffer size. By default, the lookahead size is 0, in which case the codec will behave according to the default. When a non-zero lookahead value is specified, then frame encoding is delayed as long as the lookahead buffers are filled with so many input frames. By one form, only the actual encoding of pictures is delayed but the frame type decision continues as in the default case with the Pdistance module. The basic idea in this rate control mode is to foresee the temporal complexity (and motion data) of all the future frames in the lookahead buffer before deciding the quantizer for the current frame. In this way, the system considers both the past as well as future complexity of the scene (so far as the lookahead buffers allow) before deciding the frame quantizer.

The pre-analysis includes a selection of lookahead or future frames for pre-analysis and that are to be used to analyze a current frame being analyzed and coded. The lookahead can take any value between 0 and 250 lookahead frames by one example, and it will increase the input-output lag by so many frames. The number of lookahead frames may also be as little as 1-4 and as much as 30 or 120 by other examples. The number of lookahead frames selected may depend on the application (whether real-time or live such as with a video conference, or slight delay such as a live tv show, or non-live such as for a DVD recording), where the closer to real-time, the smaller the number of lookahead frames will be selected for good quality video. Also, the system buffer size (whether at an encoder or at a decoder) may limit the number of frames that can be held.

Lookahead refers to future frames that, by one form, are pre-analyzed for determining what sort of quantizer (Qp) should be used for the current frame (and may be referred to as a group of reference frames that are looked to). Thus, lookahead simply refers to how many frames are looked into the future so that knowing something about the future (that could for instance tell us if future frames are similar in motion, i.e., scene is moving slowly or fast) that could help us determine how accurately (what quantizer value/quality level) to encode the current frame. For instance, if some part of a scene/video object in a frame is slow moving (persistent) then that area of the scene (such as frames) should be coded with high quality since the frame may be re-used (copied) to other frames without having to update it. On the other hand, if part of a frame/scene is fast moving, it may not warrant coding at the highest quality since persistence will be short and the scene will go out of range quickly.

By one specific example, if frame number 4 is processing in a sequence of frames such that frame 0 was the first frame. Assume for a 30 frame lookahead, then for coding of frame 4, frames up to frame number 34 are read and these 30 additional lookahead frames are used to determine the extent of persistence in a scene that could lead to a determination of a spatially varying block/tile based quantizer map (different from but related to picture quantizer) that would provide the highest encoding quality. On the other hand, without the 30 future lookahead frames, it may not have been possible to determine the future persistence, and the current frame may have resulted in the use of a moderate quantizer (instead of a finer quantizer) for a certain area even though the quality could be better.

The pre-analysis may include determining the complexity of each lookahead frame as well as motion characteristics, such as object detection so forth to determine a current frame's global

picture based quantizer as well as local block based quantizer map to get the best quality while factoring in whether or not the frame, or a section of the frame, is likely to reused as a reference or for copying to other frames.

Regarding the Pdistance process as such, there is no change in the pdistance logic, so that frame type and rank decisions will happen as explained above. As mentioned, only the frame encoding is delayed while a lookahead number of pictures are accumulated in the lookahead buffers. An additional latency parameter is associated with each input frame which is set to true in display order for a subgroup of pdistance frames at a time, and for which the frame type decision already occurred when the lookahead buffer count exceeds the lookahead value. The frame encoding decision may depend on this latency value. When the sequence reaches an end of file, this latency value is ignored in the frame encoding decisions.

After the rate control startup is performed, picture rate control with lookahead can be performed 2303. As with picture rate control operation 703, this is the frame rate control before encoding of each frame which determines the current frame Qp.

Also, process 2300 includes other operations the same or similar to those in process 700, and already adequately described in process 700, such as calculate average temporal (and spatial) complexity (TSC) 2304 (albeit based on the input of future frames into lookahead buffers), calculate QP limits 2306, and the parts of the frame determination 2314 that do not use the buffer status such as the operations for sequence start (2322 to 2326) and blank or static scenes (operations 2216 to 2220). The description of these operations need not be repeated here.

Once it is determined that the frame sequence to be analyzed is not a beginning of a scene, and is not a blank or static scene, operation 2300 proceeds with provide frame Qp based on buffer status with lookahead 2327, which in one example may be performed by the buffer control 412 as described herein. Here, buffer control is the same or similar as in the previous algorithms except for the derivation of the complexity delta and the analysis of select future frames in the lookahead buffer as explained below. For review, by default the current mean operating Qp should be used for coding the current frame if the current buffer status is close to the operating point, and the complexity of the current frame is similar to the average complexity so far. Otherwise, a delta Qp is derived that may be applied to the current mean Qp which will take the buffer close to the operating point. This is derived as follows.

Before encoding of each frame, buffer control may use a buffer model 410, and in this case which is a lookahead buffer model, and a buffer fullness unit 414, by one example, to project the buffer status if the current frame is encoded with a Qp derived from the current mean Pqp (target ZERO_RANK P-picture quantizer (Qp) that provides the ideal target bit rate) using equation (4) depending on its type and rank. Depending on the estimated buffer status, the current Pqp is either left unmodified or modified so as to avoid buffer overflow or underflow. The buffer control tries to modulate the current Pqp to keep it close to the target Pqp, whenever there is buffer fluctuation due to variation in scene complexity. So, keeping the buffer at the mean operating point (buf_mop) is equivalent to maintaining the Pqp close to a mean Qp operating point for a scene of given complexity. Whenever Pqp is modified by the buffer control, all of the Qps corresponding to different frame types and ranks are updated using equation (4). The buffer control works as follows.

Referring to FIG. 20 again, the buffer initialization 2329 is not repeated here since it is similar to that already described above with buffer initialization 730 and equations (27) to (30) and (32), except that average buffer fullness (avg_buf_ful) may not need to be determined here.

The process 2300 continues with lookahead buffer estimation 2331 of buffer bits. Here, the process is the same or similar as that for process 700 and bit estimation 732 except that instead of an SGOP, the process applies for all the frames available in the lookahead buffers and whose coding type is decided, and including the future frames in the lookahead buffer. As before, bits are estimated (fr_est_bits) by determining if a history exists 2334, and if not using the bit ratio estimation method 2336, and if so, the linear estimation method 2338 that are both used for initial (sequence beginning/start) frames, and depending on the Qp derived from the current Pqp. The modification relevant here is to apply qp_deltaf factors derived below during temporal adaptive control used for the complexity delta for each frame to estimate the bits. For those frames which are already coded in the current frame sequence, their actual bits are already updated in the buffer. The purpose of the buffer estimation then, with the current state of buffers, Qps, and estimation models, is to project the buffer status and the buffer increment if all the future frames available in the lookahead buffers were to be coded using the current operating set of Qps. The bit prediction 2340 provided by the estimation is then available for buffer increment and buffer estimation calculation.

The buffer increment and estimated buffer is calculated 2342 as follows where buffer increment buf_incr = 0, for all frames in the lookahead buffer,

$$\text{buf\_incr} \mathrel{+}= (\text{fr\_est\_bits} - \text{target\_rate}) \quad (98)$$

where fr_est_bits is the estimated bits for the current frame using the bits ratio method or the linear estimation method of the start sequence frame Qp process as mentioned before, and where the target_rate is the target bits per frame, and with its temporal spatial complexity and the Qp corresponding to the current frame type and rank derived from the current Pqp using equation (4).

If the current buffer fullness is buf_ful, then the projected buffer status (buf_est), if we continue with the same Pqp, is:

$$\text{buf\_est} = (\text{buf\_ful} + \text{buf\_incr}) \quad (99)$$

This shows the buffer status if all the frames in the lookahead buffers are coded using the same operating mean Pqp.

As with process 700, once the bit prediction is obtained for the buffer, and the buffer levels, increments, and thresholds are established, process 2300 may continue with determine 2344 delta Qp, performed by, in one example, a delta Qp unit 416, in order to set a maximum adjustment of the Qp. To review in more detail, a mean operating point is selected in the buffer initially. For example in equation (27), buf_full is set to buf_mop which is about 40% level of the total buffer (buf_max). Whenever the current buffer fullness (buf_ful) deviates from this operating point (buf_mop), the effort should be to bring it back to the operating point slowly. We can assume that corresponding to the mean operating point of the buffer (buf_mop), there will be a mean operating point for Pqp, i.e., Pqp_mop. So the objective of the buffer control is to maintain the Pqp close to this operating point, and whenever the Pqp deviates significantly from the operating point due to change in scene complexity, the Pqp may be modulated in small amounts to reach the operating point.

However bigger quantizer steps between neighboring frames will cause visual changes in quality. So the effort should be to bring Pqp to the target operating point in small delta values frame to frame. Thus, the delta value is a limit on how much Qp can change from frame to frame. If buf_full is the current buffer fullness, buf_est is the estimated buffer level, and buf_incr is the increment in the buffer, then the delta_qp for vbv can be derived as follows.

$$\text{Pqp}' = \text{Pqp} + \text{delta\_qp} \quad (100)$$

where,

$$\text{delta\_qp} = \text{Pqp} * (\text{vbv\_delta\_qpf} - 1.0) \quad (101)$$

The delta Qp weight factor (vbv_delta_qpf) is due to buffer deviation. This is the delta_qpf corresponding to the current buffer status. Thus, instead of comparing a buffer increment to a threshold as with the one-pass system without lookahead explained above, here the vbv_delta_qpf is determined based on the difference between a temporary current buffer **based on the bitrate of frames encoded so far** and an error calculation. This current buffer 2345 is computed as follows. A temporary buffer is initialized:

$$buf = 0.25 * bitrate \quad (102)$$

$$buf \mathrel{*}= MAX(\, sqrt\left((1 - \left(\frac{(enc\_order+1)}{total\_frames}\right)) * 32.0\right), 0.50) \quad (103)$$

where enc_order is the encoding order number of the current frame, and total_frames is the total number of pictures to be encoded. The vbv delta may then be calculated 2347 by the vbv unit 420 by one example, as follows:

$$tgt\_err = \text{buf\_est} - \text{buf\_mop} \quad (104)$$

$$vbv\_delta\_qpf = 1.0/MIN(MAX\left(\frac{buf - tgt\_err}{buf}, 0.85\right), 1.25) \quad (105)$$

where tgt_err is the target error given as the difference between the total target bits and the estimated bits including the previous target error due to past frames encoded so far. The modified Pgp' may be calculated 2349 according to equation (100).

The modified Pqp' is then subjected to Qp limits 2351 as described with operation 706 and 2306, to avoid too much deviation between successive frames.

$$\text{Pqp}' = \text{clip}(\text{Pqp}', \text{min\_qp}, \text{max\_qp}) \quad (106)$$

Thereafter, a Qp update 2353 may be performed whenever P Qp is modified at a frame so that the Qps corresponding to different ranks are updated using equation (4). The Qp corresponding to the current frame type and rank is assigned as est_qp for the current frame which is used for the actual coding of the frame.

After these adjustments are made, then the complexity delta may be added to the latest Qp. Specifically, the complexity delta (cplx_delta_qpf) may be calculated by the cmplx unit 422 by one example, and may include lookahead temporal adaptive quantization 2355 performed by a control 424 by one example. Similar to the idea of temporal adaptive quantization used in two-pass VBR rate control (process 2200), the process 2300 can incorporate the adaptive quantization into one-pass CBR using the complexities of future frames in the lookahead buffers. Key frame boosting and temporal complexity masking are incorporated using adaptive quantization over the chosen frame Qp. Key frame boosting and frame complexity variations are mapped to a delta Qp factor which can be applied to a current frame Qp. The temporal adaptive quantization is applied on the current frame using delta_qpf derived as follows.

A lookahead key frame boosting unit 426 may perform key frame boosting 2357 in low complex or static scenes. As explained above for process 2200, for key frames in low complex scenes, preserving quality by spending extra bits would save the bits of other frames. The scene complexity is detected based on the complexity proportions and factors in complexity of the lookahead frames, and a Qp weighting factor is derived for the I-picture. The key_scale_factor is derived similar to the previous section (equations (62) and (63)). For each frame j following an I-picture k,

$$\text{if fs->TSC} < (0.5*\text{width}*\text{height}),\ \text{key\_scale\_factor} \mathrel{+}= 0.005 * (j - k) \tag{107}$$

where TSC is temporal spatial complexity of the frame being analyzed. However, the frame is mapped to a delta Qp factor as follows.

$$\text{qp\_deltaf[i]} = 1.0/\text{key\_scale\_factor} \tag{108}$$

A lookahead temporal complexity masking unit 428 may perform temporal complexity masking (or just complexity masking) 2359 which may compare the current frame complexity to the average complexity of all the frames available in the lookahead buffers. The temporal weighting factors for the rest of all of the frames available in the lookahead buffers are derived using relative proportions of complexity compared to average complexity of all the available frames so far. First, average complexity is computed of all the frames in the lookahead avgTSC. The delta Qp factor for each frame is computed as the relative proportion of the frames complexity to the average complexity.

For each frame(i) in the sequence available in the lookahead buffers,

if fs[i].TSC < avgTSC,

$$\text{qp\_deltaf[i]} = \text{MAX}(0.85, \text{pow}\left(\frac{\text{fs[i].TSC}}{\text{avgTSC}}, \frac{1.0}{\text{MAX}(3.0,r+3)}\right)) \quad (109)$$

if fs[i] .TSC > avgTSC,

$$\text{qp\_deltaf[i]} = \text{MAX}(1.20, \text{pow}\left(\frac{\text{fs[i].TSC}}{\text{avgTSC}}, \frac{1.0}{\text{MAX}(2.0,r+2)}\right)) \quad (110)$$

Here the temporal complexities are mapped to Qp delta factors which can be used as a scaling factor for the frame Qp.

A lookahead Qp factor smoothing unit 430 may be used to perform Qp factor smoothing 2361. The Qp factors for successive frames available in the lookahead buffers are smoothed out using a simple Gaussian filter to avoid wide variation in Qp between successive frames which can lead to rapid variations in quality. These delta Qp factors are applied on the frame Qp derived as follows. For a chosen sigma,

$$\text{filter\_size} = 1.0 + (\text{int})(\text{sigma} * 4.0) \quad (111)$$

where sigma is chosen as 1.5 to use a 3 taps on either side of the current frame. For a given frame, the filter is applied on a series of samples with (filter_size/2) samples on either side.

$$\text{coeff} = \exp(-\text{d} * \text{d}/(\text{sigma} * \text{sigma})) \quad (112)$$

where d is the index of the current sample with regard to the reference sample.

$$\text{q+= qp\_deltaf[d]} * \text{coeff} \quad (113)$$

$$\text{sum+= coeff} \quad (114)$$

where q and sum are initialized to zero for each reference sample. Final qp_deltaf is derived as

$$\text{qp\_deltaf[i]} = \text{q/sum} \quad (115)$$

To then calculate 2363 the frame Qp, these delta Qp factors calculated for each frame are used in the following equations, which may be performed by a sequence non-start/active scene Qp calculation unit 418 in one example. If est_qp is the estimated Qp for the current frame from vbv calculations and adjusted for Qp limits and rank (equation (106) and (4)), then

$$\text{delta\_qp} = \text{est\_qp} * (\text{cplx\_delta\_qpf} - 1.0) \quad (116)$$

$$\text{Pqp}'' = \text{Pqp}' + \text{delta\_qp} \quad (117)$$

$$\text{Pqp}'' = \text{clip}(\text{Pqp}', \text{min\_qp}, \text{max\_qp}) \quad (118)$$

where the he modified frame Qp (Pqp'') is again subject to new limit Qps 2358. This Qp boundary check is effectively the second boundary check for the Buffer status sensitive Pqp'', while this operation may also include the boundary check for the start, static, or blank frames. The Qp limits are applied as described above with operation 706 and 2306 for example to avoid too much deviation between successive frames. The Qp is then available for frame encoding.

A rate control (RC) update 2360 may be performed after encoding of each frame is complete, so that the coding state and the buffer status are updated. If the current frame of specific type and rank with complexity cplx is encoded using act_qp, and it produces act_bits, then the buffer and estimation models parameters are updated. For the buffer update,

$$\text{Buffer Update: } \text{buf\_ful} \mathrel{+}= (\text{act\_bits} - \text{target\_rate}) \quad (119)$$

For the estimation models update, the linear estimation model that uses bit estimators corresponding to the current frame type and rank is updated as with the sequence start estimation models including equations (23) – (26).

An end of field (EOF) check is performed 2362, and if it is not the end of the field signifying the end of a GOP, frame sequence, video, or other predetermined length, then the process is performed again starting with the TSC calculation 2306. If the EOF is reached, an RC shut down 2364 is performed that clears the rate control parameters.

One-pass CBR with Lookahead Algorithm Summary

This section describes the algorithm flow in brief generally following FIG. 23.

1. RC Init: Initialize the RC parameters

2. RC Start Up: Based on the command line parameters, set different rate control parameters (buffers, estimation models etc.) to appropriate values. Based on the bitrate and the resolution of the sequence, initial Qps are derived based on a predefined table.

3. Pdistance with lookahead: Frame type and rank decisions will happen as earlier and a latency parameter is set depending on the number of pictures available in the lookahead buffers. Frame encoding is allowed only when corresponding latency value is set.

4. Pic Rate Control with Lookahead: This is the frame rate control before encoding of each frame which decides the current frame Qp.

A. Avg Complexity: Based on all the input frames that are available in the lookahead buffers, compute average spatio-temporal complexity.

B. Check for Static/Blank scenes: If the scene is static or a series of blank frames, then the default fixed Qp is used. Even if this happens in the middle of a scene (for instance when the scene becomes static/blank for a short time), then the last stable state Qps are used for all frames so far they are detected as blank/static frames.

C. Qp limits for the frame:

i. Beginning of a sequence: If it's the absolute beginning of a sequence, then the initial Qp estimation table developed based on experiments is used to derive the boundary Qps. This table is a map of complexity, bitrate (bpp) and ZERO_RANK frame Qp which tries to predict the mean ZERO_RANK P-picture Qp for the complete sequence with the calculated average complexity which can produce the specified target bitrate. Qp limits are also derived from corresponding predefined tables.

ii. After some initial history is built: Since we have some history of past total bits consumed for the average complexity and the average Qp used, we try to predict the Qp for the new frame based on the projected complexity and bits including the current frame relative to the past statistics which can take the total bit consumption including the current frame close to the target rate and the Qp limits are derived from this Qp using the standard P to I and P to F Qp mapping.

iii. Default: The case above (ii) is only used when the current frame complexity and the actual bitrate so far are relatively close to the average complexity so far and the target bitrate respectively. In any other case, the default Qp is going to be the average Qp so far and the Qp limits are derived from this Qp using the standard P to I and P to F Qp mapping.

D. Temporal Adaptive Quantization: Based on the complexity of all the input frames available in the lookahead buffers, Adaptive quantization is applied over frame Qp using the delta Qp factors derived from Key frame boosting and (temporal) complexity masking.

i. Key Frame boosting in low complex or static scenes: For key frames in low complex scenes preserving the quality by spending extra bits would save the bits of other frames. The scene complexity is detected based on the complexity proportions and a Qp weighting factor is derived for the I-picture.

ii. Temporal Complexity masking: Similarly the temporal weighting factors for rest of all the frames available in the lookahead buffers are derived using relative proportions of complexity compared to average complexity of all the available frames so far.

iii. Qp factors smoothing: The Qp factors for successive frames available in the lookahead buffers are smoothed out using a simple Gaussian filter to avoid wide variation in Qp between successive frames which can lead to rapid variations in quality. These delta Qp factors are applied on the frame Qp derived as follows.

E. Frame Qp:

i. Beginning of a sequence or Scene Change: The frame Qp is derived based on the average complexity, target bitrate (bpp) and Qp modeling developed based on lot of experiments. This Qp is constrained to the limits derived in the previous section C. This is necessary because sometimes the model may predict wrong Qps due to abnormal complexity in the beginning (blank frames etc.).

ii. Buffer Control: For any other frame, buffer control decides the frame Qp. By default the current mean operating Qp should be used for coding the current frame if the current buffer status is close to the operating point and the complexity of the current frame is similar to the average complexity so far. Otherwise, we would like to derive a delta Qp that needs to be applied to the current mean Qp which will take the buffer close to the operating point. This is derived in a series of steps as follows.

a. Buffer Estimation & Buffer Increment: With the current state of buffers, Qps and estimation models, project the buffer status and the buffer increment if all the future frames available in the lookahead buffers were to be coded using the current operating set of Qps.

(i) History Available: If sufficient history is built in to the bit estimation models, these can be used to estimate the bits a frame is going to consume for its complexity and the Qp corresponding to the type and rank of the current frame.

(ii) No History: Otherwise, some proportion models derived from experiments are used to estimate the frame bits with the given complexity and Qp. These bit proportion models include I to P bit ratios, P to F bit ratios etc.

b. Delta Qp factors Calculation: Two factors of delta Qp are derived, one corresponding to the current state of the buffer, the other one corresponding to the current scene complexity.

(i) VBV Delta: The delta Qp weight factor due to buffer deviation is calculated.

(ii) Complexity Delta: The delta Qp weight factor derived in d (iii) is used as complexity delta Qp factor.

c. Frame Qp Calculation: The new P Qp is derived by adding the above VBV delta Qp to the current mean Qp, and limiting to the boundary Qps derived in Section C. The different frame type and rank Qps are re-calculated based on this new P Qp. Finally the complexity delta Qp is also added to the current frame Qp and once again limited to the boundary Qps derived in Section C. This gives the final frame Qp.

5. Pic RC Update: This updates the buffer status, estimation models at the end of each frame encode, based on the current frame complexity, Qp and bits.

6. RC Shut down: clears up the rate control parameters.

Changes to support Low Delay Coding

By another aspect, lower GOP sizes and IPPP coding are implemented where a set of changes are added to the latest NGV encoder to support low delay coding for applications like video conferencing. The typical configuration for low delay coding is (1) IPPP coding, and (2) buffer size of a few frames compared to about a second or two buffer delay in the normal case. The one-pass CBR rate control with lookahead buffer control algorithm is able to handle the low delay coding scenario due to the following two reasons:

(a) The rate control works very generally; it doesn't assume any fixed GOP sizes or pyramid structures or frame types. It uses the prediction models to estimate bits of each frame depending on whatever type or rank is assigned to it by the pdistance module. Such estimated bits for all the future frames available is accumulated and compensated on the current frame with the error with regard to the target bits. So this rate control can work with any configuration: any frame types, any GOP sizes or any pyramid structures.

(b) By default, two second buffer (2*bitrate) is assumed for buffer size in rate control if the buffer size is not specified. If a reduced buffer size is desired for any application, that can be specified using the parameter "-bs bufferSizeinKb". This will set the maximum buffer fullness to this value and buffer control operates within this limit.

When buffer size is specified in the command line, tight rate control is enabled. Generally, the frame Qp is modified by looking at buffer change over a large number of future frames allowing a little fluctuation in bits for individual frames. This is necessary because with F pyramid coding, the bits vary quite rapidly frame to frame, and to avoid corresponding rapid Qp variations, this scheme is adapted as disclosed herein. However, in case of tight rate control, the buffer status is checked for each frame individually, and frame Qp is compensated based on the estimated buffer status.

To accomplish reduced (and tight) rate control, frame Qp compensation based on estimated buffer status, the following equations are used. After the frame Qp is decided as described above in each control rate process for each frame, if est_qp and est_bits are the estimated Qp and bits for the current frame,

$$\text{buf\_est} = \text{buf\_ful} + (\text{est\_bits} - \text{target\_rate}) \quad (120)$$

$$\text{tgt\_err} = (\text{buf\_est} - \text{buf\_mop}) \quad (121)$$

$$\text{Vbv\_buf} = (2.0 * \text{buf\_size}) \quad (122)$$

$$\text{est\_qp} \mathrel{*}= \text{MIN}(\text{MAX}((\text{vbv\_buf} - \text{tgt\_err})/\text{vbv\_buf}, 0.75), 1.50) \quad (123)$$

Various components of the systems described herein may be implemented in software, firmware, and/or hardware and/or any combination thereof. For example, various components of system 300 may be provided, at least in part, by hardware of a computing System-on-a-Chip (SoC) such as may be found in a computing system such as, for

example, a smart phone. Those skilled in the art may recognize that systems described herein may include additional components that have not been depicted in the corresponding figures. For example, the systems discussed herein may include additional components such as bit stream multiplexer or de-multiplexer modules and the like that have not been depicted in the interest of clarity.

Processes 700, 2200, and 2300 may be implemented via any of the coder systems as discussed herein. Further, the processes may be repeated either in serial or in parallel on any number of instantiations of video data such as prediction error data partitions, original data partitions, or wavelet data or the like.

While implementation of the example processes herein may include the undertaking of all operations shown in the order illustrated, the present disclosure is not limited in this regard and, in various examples, implementation of the example processes herein may include the undertaking of only a subset of the operations shown and/or in a different order than illustrated.

Various components of the systems described herein may be implemented in software, firmware, and/or hardware and/or any combination thereof. For example, various components of system 1400 may be provided, at least in part, by hardware of a computing System-on-a-Chip (SoC) such as may be found in a computing system such as, for example, a smart phone. Those skilled in the art may recognize that systems described herein may include additional components that have not been depicted in the corresponding figures. For example, the systems discussed herein may include additional components such as bit stream multiplexer or de-multiplexer modules and the like that have not been depicted in the interest of clarity.

In addition, any one or more of the operations discussed herein may be undertaken in response to instructions provided by one or more computer program products. Such program products may include signal bearing media providing instructions that, when executed by, for example, a processor, may provide the functionality described herein. The computer program products may be provided in any form of one or more machine-readable media. Thus, for example, a processor including one or more processor core(s) may undertake one or more of the operations of the example processes herein in response to program code and/or instructions or instruction sets conveyed to the processor by one or more machine-readable media. In general, a machine-readable medium may convey software in the form of program code and/or instructions

or instruction sets that may cause any of the devices and/or systems described herein to implement at least portions of the video systems as discussed herein.

As used in any implementation described herein, the term "module" refers to any combination of software logic, firmware logic and/or hardware logic configured to provide the functionality described herein. The software may be embodied as a software package, code and/or instruction set or instructions, and "hardware", as used in any implementation described herein, may include, for example, singly or in any combination, hardwired circuitry, programmable circuitry, state machine circuitry, and/or firmware that stores instructions executed by programmable circuitry. The modules may, collectively or individually, be embodied as circuitry that forms part of a larger system, for example, an integrated circuit (IC), system on-chip (SoC), and so forth. For example, a module may be embodied in logic circuitry for the implementation via software, firmware, or hardware of the coding systems discussed herein.

FIG. 24 is an illustrative diagram of example video coding system 2400, arranged in accordance with at least some implementations of the present disclosure. In the illustrated implementation, video coding system 2400 may include imaging device(s) 2401, video encoder 100, video decoder 200 (and/or a video coder implemented via logic circuitry 2450 of processing unit(s) 2420), an antenna 2402, one or more processor(s) 2403, one or more memory store(s) 2404, and/or a display device 2405.

As illustrated, imaging device(s) 2401, antenna 2402, processing unit(s) 2420, logic circuitry 2350, video encoder 100, video decoder 200, processor(s) 2403, memory store(s) 2404, and/or display device 2405 may be capable of communication with one another. As discussed, although illustrated with both video encoder 100 and video decoder 200, video coding system 2400 may include only video encoder 100 or only video decoder 200 in various examples.

As shown, in some examples, video coding system 2400 may include antenna 2402. Antenna 2402 may be configured to transmit or receive an encoded bitstream of video data, for example. Further, in some examples, video coding system 2400 may include display device 2405. Display device 2405 may be configured to present video data. As shown, in some examples, logic circuitry 2450 may be implemented via processing unit(s) 2420. Processing unit(s) 2420 may include application-specific integrated circuit (ASIC) logic, graphics processor(s), general purpose processor(s), or the like. Video coding system 2400 also may include optional processor(s) 2403, which may similarly include application-specific integrated

circuit (ASIC) logic, graphics processor(s), general purpose processor(s), or the like. In some examples, logic circuitry 2450 may be implemented via hardware, video coding dedicated hardware, or the like, and processor(s) 2403 may implemented general purpose software, operating systems, or the like. In addition, memory store(s) 2404 may be any type of memory such as volatile memory (e.g., Static Random Access Memory (SRAM), Dynamic Random Access Memory (DRAM), etc.) or non-volatile memory (e.g., flash memory, etc.), and so forth. In a non-limiting example, memory store(s) 2404 may be implemented by cache memory. In some examples, logic circuitry 2450 may access memory store(s) 2404 (for implementation of an image buffer for example). In other examples, logic circuitry 2450 and/or processing unit(s) 2420 may include memory stores (e.g., cache or the like) for the implementation of an image buffer or the like.

In some examples, video encoder 100 implemented via logic circuitry may include an image buffer (e.g., via either processing unit(s) 2420 or memory store(s) 2404)) and a graphics processing unit (e.g., via processing unit(s) 2420). The graphics processing unit may be communicatively coupled to the image buffer. The graphics processing unit may include video encoder 100 as implemented via logic circuitry 2450 to embody the various modules as discussed with respect to FIG. 1 and/or any other encoder system or subsystem described herein. For example, the graphics processing unit may include coding partitions generator logic circuitry, adaptive transform logic circuitry, content pre-analyzer, encode controller logic circuitry, adaptive entropy encoder logic circuitry, and so on. The logic circuitry may be configured to perform the various operations as discussed herein.

In one example, graphics processing unit of video encoder 100 may be configured to obtain frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames. This hierarchy may also include at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, where P-pictures use past frames relative to the order as references, and where pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both. The method may also include determining a quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of at least one current frame, and where each frame is given a rank associated with the level the frame is on.

Video decoder 200 may be implemented in a similar manner as implemented via logic circuitry 2450 to embody the various modules as discussed with respect to decoder 200 of FIG. 2 and/or any other decoder system or subsystem described herein.

In some examples, antenna 2402 of video coding system 2400 may be configured to receive an encoded bitstream of video data. As discussed, the encoded bitstream may include data associated with a coder that has an image buffer to hold frame data and a graphics processing unit performing the method disclosed above. Video coding system 2400 may also include video decoder 200 coupled to antenna 2402 and configured to decode the encoded bitstream.

In embodiments, features described herein may be undertaken in response to instructions provided by one or more computer program products. Such program products may include signal bearing media providing instructions that, when executed by, for example, a processor, may provide the functionality described herein. The computer program products may be provided in any form of one or more machine-readable media. Thus, for example, a processor including one or more processor core(s) may undertake one or more features described herein in response to program code and/or instructions or instruction sets conveyed to the processor by one or more machine-readable media. In general, a machine-readable medium may convey software in the form of program code and/or instructions or instruction sets that may cause any of the devices and/or systems described herein to implement at least portions of the features described herein.

FIG. 25 is an illustrative diagram of an example system 2500, arranged in accordance with at least some implementations of the present disclosure. In various implementations, system 2500 may be a media system although system 2500 is not limited to this context. For example, system 2500 may be incorporated into a personal computer (PC), laptop computer, ultra-laptop computer, tablet, touch pad, portable computer, handheld computer, palmtop computer, personal digital assistant (PDA), cellular telephone, combination cellular telephone/PDA, television, smart device (e.g., smart phone, smart tablet or smart television), mobile internet device (MID), messaging device, data communication device, cameras (e.g. point-and-shoot cameras, super-zoom cameras, digital single-lens reflex (DSLR) cameras), and so forth.

In various implementations, system 2500 includes a platform 2502 coupled to a display 2520. Platform 2502 may receive content from a content device such as content services device(s) 2530 or content delivery device(s) 2540 or other similar content sources. A navigation controller 2550 including one or more navigation features may be used to interact with, for

example, platform 2502 and/or display 2520. Each of these components is described in greater detail below.

In various implementations, platform 2502 may include any combination of a chipset 2505, processor 2510, memory 2512, antenna 2513, storage 2514, graphics subsystem 2515, applications 2516 and/or radio 2518. Chipset 2505 may provide intercommunication among processor 2510, memory 2512, storage 2514, graphics subsystem 2515, applications 2516 and/or radio 2518. For example, chipset 2505 may include a storage adapter (not depicted) capable of providing intercommunication with storage 2514.

Processor 2510 may be implemented as a Complex Instruction Set Computer (CISC) or Reduced Instruction Set Computer (RISC) processors, x86 instruction set compatible processors, multi-core, or any other microprocessor or central processing unit (CPU). In various implementations, processor 2510 may be dual-core processor(s), dual-core mobile processor(s), and so forth.

Memory 2512 may be implemented as a volatile memory device such as, but not limited to, a Random Access Memory (RAM), Dynamic Random Access Memory (DRAM), or Static RAM (SRAM).

Storage 2514 may be implemented as a non-volatile storage device such as, but not limited to, a magnetic disk drive, optical disk drive, tape drive, an internal storage device, an attached storage device, flash memory, battery backed-up SDRAM (synchronous DRAM), and/or a network accessible storage device. In various implementations, storage 2514 may include technology to increase the storage performance enhanced protection for valuable digital media when multiple hard drives are included, for example.

Graphics subsystem 2515 may perform processing of images such as still or video for display. Graphics subsystem 2515 may be a graphics processing unit (GPU) or a visual processing unit (VPU), for example. An analog or digital interface may be used to communicatively couple graphics subsystem 2515 and display 2520. For example, the interface may be any of a High-Definition Multimedia Interface, DisplayPort, wireless HDMI, and/or wireless HD compliant techniques. Graphics subsystem 2515 may be integrated into processor 2510 or chipset 2505. In some implementations, graphics subsystem 2515 may be a stand-alone device communicatively coupled to chipset 2505.

The graphics and/or video processing techniques described herein may be implemented in various hardware architectures. For example, graphics and/or video functionality may be integrated within a chipset. Alternatively, a discrete graphics and/or video processor may be used. As still another implementation, the graphics and/or video functions may be provided by a general purpose processor, including a multi-core processor. In further embodiments, the functions may be implemented in a consumer electronics device.

Radio 2518 may include one or more radios capable of transmitting and receiving signals using various suitable wireless communications techniques. Such techniques may involve communications across one or more wireless networks. Example wireless networks include (but are not limited to) wireless local area networks (WLANs), wireless personal area networks (WPANs), wireless metropolitan area network (WMANs), cellular networks, and satellite networks. In communicating across such networks, radio 2518 may operate in accordance with one or more applicable standards in any version.

In various implementations, display 2520 may include any television type monitor or display. Display 2520 may include, for example, a computer display screen, touch screen display, video monitor, television-like device, and/or a television. Display 2520 may be digital and/or analog. In various implementations, display 2520 may be a holographic display. Also, display 2520 may be a transparent surface that may receive a visual projection. Such projections may convey various forms of information, images, and/or objects. For example, such projections may be a visual overlay for a mobile augmented reality (MAR) application. Under the control of one or more software applications 2516, platform 2502 may display user interface 2522 on display 2520.

In various implementations, content services device(s) 2530 may be hosted by any national, international and/or independent service and thus accessible to platform 2502 via the Internet, for example. Content services device(s) 2530 may be coupled to platform 2502 and/or to display 2520. Platform 2502 and/or content services device(s) 2530 may be coupled to a network 2560 to communicate (e.g., send and/or receive) media information to and from network 2560. Content delivery device(s) 2540 also may be coupled to platform 2502 and/or to display 2520.

In various implementations, content services device(s) 2530 may include a cable television box, personal computer, network, telephone, Internet enabled devices or appliance capable of delivering digital information and/or content, and any other similar device capable of

unidirectionally or bidirectionally communicating content between content providers and platform 2502 and/display 2520, via network 2560 or directly. It will be appreciated that the content may be communicated unidirectionally and/or bidirectionally to and from any one of the components in system 2500 and a content provider via network 2560. Examples of content may include any media information including, for example, video, music, medical and gaming information, and so forth.

Content services device(s) 2530 may receive content such as cable television programming including media information, digital information, and/or other content. Examples of content providers may include any cable or satellite television or radio or Internet content providers. The provided examples are not meant to limit implementations in accordance with the present disclosure in any way.

In various implementations, platform 2502 may receive control signals from navigation controller 2550 having one or more navigation features. The navigation features of controller 2550 may be used to interact with user interface 2522, for example. In various embodiments, navigation controller 2550 may be a pointing device that may be a computer hardware component (specifically, a human interface device) that allows a user to input spatial (e.g., continuous and multi-dimensional) data into a computer. Many systems such as graphical user interfaces (GUI), and televisions and monitors allow the user to control and provide data to the computer or television using physical gestures.

Movements of the navigation features of controller 2550 may be replicated on a display (e.g., display 2520) by movements of a pointer, cursor, focus ring, or other visual indicators displayed on the display. For example, under the control of software applications 2516, the navigation features located on navigation controller 2550 may be mapped to virtual navigation features displayed on user interface 2522. In various embodiments, controller 2550 may not be a separate component but may be integrated into platform 2502 and/or display 2520. The present disclosure, however, is not limited to the elements or in the context shown or described herein.

In various implementations, drivers (not shown) may include technology to enable users to instantly turn on and off platform 2502 like a television with the touch of a button after initial boot-up, when enabled, for example. Program logic may allow platform 2502 to stream content to media adaptors or other content services device(s) 2530 or content delivery device(s) 2540 even when the platform is turned "off." In addition, chipset 2505 may include hardware and/or software support for 5.1 surround sound audio and/or high definition 7.1 surround sound audio,

for example. Drivers may include a graphics driver for integrated graphics platforms. In various embodiments, the graphics driver may comprise a peripheral component interconnect (PCI) Express graphics card.

In various implementations, any one or more of the components shown in system 2500 may be integrated. For example, platform 2502 and content services device(s) 2530 may be integrated, or platform 2502 and content delivery device(s) 2540 may be integrated, or platform 2502, content services device(s) 2530, and content delivery device(s) 2540 may be integrated, for example. In various embodiments, platform 2502 and display 2520 may be an integrated unit. Display 2520 and content service device(s) 2530 may be integrated, or display 2520 and content delivery device(s) 2540 may be integrated, for example. These examples are not meant to limit the present disclosure.

In various embodiments, system 2500 may be implemented as a wireless system, a wired system, or a combination of both. When implemented as a wireless system, system 2500 may include components and interfaces suitable for communicating over a wireless shared media, such as one or more antennas, transmitters, receivers, transceivers, amplifiers, filters, control logic, and so forth. An example of wireless shared media may include portions of a wireless spectrum, such as the RF spectrum and so forth. When implemented as a wired system, system 2500 may include components and interfaces suitable for communicating over wired communications media, such as input/output (I/O) adapters, physical connectors to connect the I/O adapter with a corresponding wired communications medium, a network interface card (NIC), disc controller, video controller, audio controller, and the like. Examples of wired communications media may include a wire, cable, metal leads, printed circuit board (PCB), backplane, switch fabric, semiconductor material, twisted-pair wire, co-axial cable, fiber optics, and so forth.

Platform 2502 may establish one or more logical or physical channels to communicate information. The information may include media information and control information. Media information may refer to any data representing content meant for a user. Examples of content may include, for example, data from a voice conversation, videoconference, streaming video, electronic mail ("email") message, voice mail message, alphanumeric symbols, graphics, image, video, text and so forth. Data from a voice conversation may be, for example, speech information, silence periods, background noise, comfort noise, tones and so forth. Control information may refer to any data representing commands, instructions or control words meant

for an automated system. For example, control information may be used to route media information through a system, or instruct a node to process the media information in a predetermined manner. The embodiments, however, are not limited to the elements or in the context shown or described in FIG. 25.

As described above, system 2500 may be embodied in varying physical styles or form factors. FIG. 26 illustrates implementations of a small form factor device 2600 in which system 2600 may be embodied. In various embodiments, for example, device 2600 may be implemented as a mobile computing device a having wireless capabilities. A mobile computing device may refer to any device having a processing system and a mobile power source or supply, such as one or more batteries, for example.

As described above, examples of a mobile computing device may include a personal computer (PC), laptop computer, ultra-laptop computer, tablet, touch pad, portable computer, handheld computer, palmtop computer, personal digital assistant (PDA), cellular telephone, combination cellular telephone/PDA, television, smart device (e.g., smart phone, smart tablet or smart television), mobile internet device (MID), messaging device, data communication device, cameras (e.g. point-and-shoot cameras, super-zoom cameras, digital single-lens reflex (DSLR) cameras), and so forth.

Examples of a mobile computing device also may include computers that are arranged to be worn by a person, such as a wrist computer, finger computer, ring computer, eyeglass computer, belt-clip computer, arm-band computer, shoe computers, clothing computers, and other wearable computers. In various embodiments, for example, a mobile computing device may be implemented as a smart phone capable of executing computer applications, as well as voice communications and/or data communications. Although some embodiments may be described with a mobile computing device implemented as a smart phone by way of example, it may be appreciated that other embodiments may be implemented using other wireless mobile computing devices as well. The embodiments are not limited in this context.

As shown in FIG. 26, device 2600 may include a housing 2602, a display 2604 which may include a user interface 2610, an input/output (I/O) device 2606, and an antenna 2608. Device 2600 also may include navigation features 2612. Display 2604 may include any suitable display unit for displaying information appropriate for a mobile computing device. I/O device 2606 may include any suitable I/O device for entering information into a mobile computing device. Examples for I/O device 2606 may include an alphanumeric keyboard, a numeric keypad, a

touch pad, input keys, buttons, switches, rocker switches, microphones, speakers, voice recognition device and software, and so forth. Information also may be entered into device 2600 by way of microphone (not shown). Such information may be digitized by a voice recognition device (not shown). The embodiments are not limited in this context.

While implementation of the example processes herein may include the undertaking of all operations shown in the order illustrated, the present disclosure is not limited in this regard and, in various examples, implementation of the example processes herein may include the undertaking of only a subset of the operations shown and/or in a different order than illustrated.

In addition, any one or more of the operations discussed herein may be undertaken in response to instructions provided by one or more computer program products. Such program products may include signal bearing media providing instructions that, when executed by, for example, a processor, may provide the functionality described herein. The computer program products may be provided in any form of one or more machine-readable media. Thus, for example, a processor including one or more processor core(s) may undertake one or more of the operations of the example processes herein in response to program code and/or instructions or instruction sets conveyed to the processor by one or more machine-readable media. In general, a machine-readable medium may convey software in the form of program code and/or instructions or instruction sets that may cause any of the devices and/or systems described herein to implement at least portions of the video systems as discussed herein.

As used in any implementation described herein, the term “module” refers to any combination of software logic, firmware logic and/or hardware logic configured to provide the functionality described herein. The software may be embodied as a software package, code and/or instruction set or instructions, and “hardware”, as used in any implementation described herein, may include, for example, singly or in any combination, hardwired circuitry, programmable circuitry, state machine circuitry, and/or firmware that stores instructions executed by programmable circuitry. The modules may, collectively or individually, be embodied as circuitry that forms part of a larger system, for example, an integrated circuit (IC), system on-chip (SoC), and so forth. For example, a module may be embodied in logic circuitry for the implementation via software, firmware, or hardware of the coding systems discussed herein.

Various implementations may be implemented using hardware elements, software elements, or a combination of both. Examples of hardware elements may include processors,

microprocessors, circuits, circuit elements (e.g., transistors, resistors, capacitors, inductors, and so forth), integrated circuits, application specific integrated circuits (ASIC), programmable logic devices (PLD), digital signal processors (DSP), field programmable gate array (FPGA), logic gates, registers, semiconductor device, chips, microchips, chip sets, and so forth. Examples of software may include software components, programs, applications, computer programs, application programs, system programs, machine programs, operating system software, middleware, firmware, software modules, routines, subroutines, functions, methods, procedures, software interfaces, application program interfaces (API), instruction sets, computing code, computer code, code segments, computer code segments, words, values, symbols, or any combination thereof. Determining whether an embodiment is implemented using hardware elements and/or software elements may vary in accordance with any number of factors, such as desired computational rate, power levels, heat tolerances, processing cycle budget, input data rates, output data rates, memory resources, data bus speeds and other design or performance constraints.

One or more aspects of at least one implementation may be implemented by representative instructions stored on a machine-readable medium which represents various logic within the processor, which when read by a machine causes the machine to fabricate logic to perform the techniques described herein. Such representations, known as "IP cores" may be stored on a tangible, machine readable medium and supplied to various customers or manufacturing facilities to load into the fabrication machines that actually make the logic or processor.

While certain features set forth herein have been described with reference to various implementations, this description is not intended to be construed in a limiting sense. Hence, various modifications of the implementations described herein, as well as other implementations, which are apparent to persons skilled in the art to which the present disclosure pertains are deemed to lie within the spirit and scope of the present disclosure.

The following examples pertain to further implementations.

By one implementation, a computer implemented method comprise obtaining frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum

level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both. The method also comprises determining a quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of at least one current frame, and wherein each frame is given a rank associated with the level the frame is on.

By another implementation, the method may also include providing, at least initially, a smaller quantization parameter for the frames the closer the level of the frame is to the base level; providing, at least initially, the base level with the smallest quantization parameter for the frames relative to all the other levels; forming an at least initial Qp of the rank (0) P-picture from a predetermined relationship between Qp and a value of average temporal complexity of available complexities of future frames per target bitrate; using at least the initial Qp of the P-picture rank (0) frames to form the Qp of the rank (0) non-P-picture frames; and using a predetermined mapping of P-picture rank (0) Qp value to rank (0) non-P-picture Qp value to form the rank (0) non-P-picture Qp value for a frame.

The method also may include determining a constant initial frame Qp for each non-zero level, wherein the constant frame Qp is established for all frames on a non-zero level regardless of the type of frame on the non-zero level; determining the Qp of other than rank (0) frames from the equation:

$$Qp_r = \text{inter\_q}[0] + ((\text{fpic\_q}[0] - \text{inter\_q}[0]) + (1.0/2.0)) \times r \quad (a)$$

where inter_q[0] is the initial Qp for the rank(0) P-pictures, fpic_q is the initial Qp to be used for the rank(0) F-pictures, and r is the rank of a current frame being analyzed, wherein an option is applying limits to the Qp value derived from at least one of: (1) for frames at a start of a sequence, limits derived from predetermined values associated with the relationship between spatio-temporal complexity and bits per pixel (bpp) based on bitrate, frame rate, and

width*height of a frame, and (2) for frames other than frames at a start of a sequence, limits for the frames based, at least in part, on a history including data from previously encoded frames of the same sequence as a frame being analyzed, and wherein (1) the mean QP of the frames encoded so far is based on: (avg_pqp) the average Pqps of all the frames encoded so far, (avg_cplx) the average spatio temporal complexity, (act_bits) the total bits consumed for all the frames encoded so far, (cplx) the complexity of the current frame, and (target_bits) the total target bits for all the frames including the current frame, and wherein the history option (2) is used when the resulting mean Qp is within a certain percentage of the average Pqp, and (3) setting the average Pqp of the frames encoded so far as the mean Qp used to determine the Qp limits, wherein Pqp is the P-picture rank (0) Qp; determining a more final frame Qp relative to the initial QP differently depending on whether the frame is at the start of a sequence, is part of a blank or static scene, or neither; determining a more final frame QP relative to the initial QP based, at least in part, on the rank of the frame; establishing a value of estimated bits for a P-picture (P_est_bits) based on, at least in part, a predetermined relationship between the initial P-picture rank (0) Qp and an I-picture to P-picture bits ratio; establishing a value of estimated bits for an F-picture (F_est_bits) based on a predetermined relationship between the initial P-picture rank (0) Qp and a P-picture to F-picture bits ratio, wherein an option is provided to use past reference frames, future reference frames, or both for an F-picture, and including providing the option to provide the F-picture with modified reference frames that are modified by a morphing technique or a synthesis technique, wherein estimation of bits of a frame are calculated with a linear model and are different for different combinations of frame type and level of the frame, wherein the frame types including at least P-picture, F-picture, B-picture, or I-picture.

The method also including determining a more final frame Qp to maintain buffer fullness toward a mean operating point of the buffer relative to the bit capacity of the buffer, wherein the buffer is a virtual buffer at an encoder that imitates the parameters of a buffer at a decoder; determining a more final Qp for a frame sensitive to buffer fullness of a buffer, comprising: estimating the bits of a frame with level-sensitive Qp models; determining whether to apply a video buffer verified (VBV) delta associated with buffer status and applied to adjust an initial Qp toward a mean operating Qp of a buffer, and applied depending on at least one of: (a) whether a rate of buffer increment is above a threshold, wherein the buffer increment is a value of the bits relative to an average bit buffer level for a frame sequence, and based on the estimating of the bits of a frame, and (b) whether a buffer estimate of the fullness of the buffer is above a threshold wherein the buffer estimate is associated with a buffer fullness value and the buffer

increment value; calculating a complexity delta associated with a weighting factor corresponding to complexity of a frame and associated with the buffer increment; and determining the more final Qp based, at least in part, on the initial Qp adjusted by the VBV delta if the threshold is surpassed, and the complexity delta; wherein the initial Qp is a rank (0) P-picture Qp; and at least one of (A) and (B) comprising:

(A) performing a single-pass analysis with lookahead comprising: forming a more final Qp for a current frame depending on, at least in part, a pre-analyzing a number of lookahead frames that are future frames relative to the current frame, and wherein the pre-analyzing comprises determining the complexity and motion characteristics of the lookahead frames without fully encoding the lookahead frames; determining a more final Qp based, at least in part, on adjusting a more initial Qp with a delta value including determining a delta value with: (1) a complexity delta associated with a complexity of a sequence having the current frame being analyzed and including the lookahead frames, and formed by an adaptive quantization comprising at least one of: key frame boosting by shifting bits by changing the total bit amount of a frame to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference, temporal complexity masking comprising shuffling bits by changing the total bit amount of a frame, from more complex to less complex frames, and delta Qp factor smoothing of delta values of successive frames, and (2) a video buffer verification (VBV) delta associated with the fullness of the buffer; and

(B) performing a two-pass analysis comprising: performing a first pass analysis that encodes frames of a sequence by using the initial Qp values to obtain the bits used and complexity of the frames analyzed; and performing a second pass analysis to determine a more final frame Qp for the frames of the sequence by adjusting the initial Qp by a delta value to maintain a QP close to a mean operating point QP corresponding to a mean operating point of a buffer holding the frames of the sequence, wherein the delta value is associated with the fullness status of a buffer and the complexity of the frame sequence being analyzed.

By another approach, a computer implemented method for one-pass rate control with lookahead for video coding, comprises obtaining frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-

pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both. The method also includes determining an initial quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of the current frame, and wherein each frame is given a rank associated with the level the frame is on; pre-analyzing a number of lookahead frames that are future frames relative to a current frame and analyzed to determine the complexity or motion characteristics or both of the lookahead frames without fully encoding the lookahead frames; and determining a more final Qp by using the results of the analysis of the lookahead frames to adjust the initial Qp of the current frame.

Such a method may also comprise determining a more final Qp based, at least in part, on determining a buffer status of a buffer that imitates the behavior of a buffer at a decoder, wherein the buffer includes space for the lookahead future frames, wherein determining a buffer status comprises modulating a current Qp for frames to be close to a target mean operating point QP that corresponds to the mean operating point of the buffer, wherein determining a more final Qp comprises adjusting an initial Qp by a delta value having at least part formed of at least one of: a video buffer verification delta associated with the fullness status of the buffer, and a complexity delta associated with the complexity of the frame sequence of the current frame being analyzed and including the lookahead frames, wherein the complexity delta is formed by performing adaptive quantization comprising temporal complexity masking that shuffles bits from less complex to more complex frames and by using an average complexity of the frame sequence that includes the complexities of the lookahead frames, wherein the complexity data is formed by performing adaptive quantization comprising key frame boosting by shifting bits in frame sequences of low complexity to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference, wherein the frame sequence is determined to be low complexity, at least in part, by using the complexities of the lookahead frames, wherein a resulting QP after a VBV delta is applied and associated with a fullness status of a buffer, adjusting the resulting QP by applying the complexity delta, wherein determining a more final Qp comprises adjusting an initial Qp by performing adaptive quantization that forms a delta to adjust the initial QP, and comprising forming the delta, at least in part, by QP factor smoothing of VBV and complexity delta values of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames.

By yet another implementation, a computer-implemented method for two-pass rate control for video coding, comprising: obtaining frames of pixel data in an input video order and

associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both.

The method also including performing a first pass analysis with initial default quantization parameter (Qp) for each frame analyzed, wherein the initial default Qps are established at least partly based on the level of the hierarchy of the current frame, and wherein each frame is given a rank associated with the level the frame is on, and performing a first pass analysis comprising encoding the frames sufficiently to determine a complexity and bits for the frames analyzed; and performing a second pass analysis comprising determining a more final quantization parameter (Qp) for individual frames and for coding comprising: performing adaptive quantization to enhance the quality of one or more frames comprising shifting bits by raising the total amount of bits for at least one frame while reducing the amount of bits for at least one other frame in the same sequence; and determining a target Qp for a frame to be used for coding and target bits for a frame for updating statistics of the first pass analysis.

The method also including wherein the first pass analysis encodes all of the frames in the sequence, wherein the initial default rank (0) P-picture is formed by using a predetermined relationship between complexity, bitrate, and Qp, the method comprising scaling the frames to match a target bits per frame, wherein performing adaptive quantization comprises key frame boosting comprising shifting bits to an I-frame and from the plurality of frames that use the I-frame as a reference, wherein performing adaptive quantization comprises temporal complexity masking including shuffling from more complex frames from less complex frames within a sequence, and depending on the same type and rank frame combinations, wherein the scaling and adaptive quantization result in a target bit frame size, and the method comprising determining estimated final Qp for coding of a frame depending on the bits and Qp of the frame determined in the first pass analysis and the target bit frame size (tgt_size), wherein performing adaptive quantization forms a delta Qp factor, and comprises QP factor smoothing of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames, the method comprising determining a more final Qp by using an iteration that factors in the frame rank and type of the frame being analyzed, wherein the

iteration also factors in the delta Qp factor of adaptive quantization, and wherein the iteration also factors in the bits per frame and Qp from the first pass analysis, wherein the iteration comprises: setting a test Qp value based on a step value; calculating an expected total bits for a chosen frame wherein the expected bits is based on, at least in part, frame rank and type of the frame being analyzed, delta Qp factor of adaptive quantization, and bits per frame and Qp from the first pass analysis; comparing the expected total bits to a target bits value as a threshold; changing the test Qp value differently depending on whether the threshold is surpassed; and restarting the process with setting a test Qp for a finite number of times.

In a further example, at least one machine readable medium may include a plurality of instructions that in response to being executed on a computing device, causes the computing device to perform the method according to any one of the above examples.

In a still further example, an apparatus may include means for performing the methods according to any one of the above examples.

The above examples may include specific combination of features. However, such the above examples are not limited in this regard and, in various implementations, the above examples may include the undertaking only a subset of such features, undertaking a different order of such features, undertaking a different combination of such features, and/or undertaking additional features than those features explicitly listed. For example, all features described with respect to the example methods may be implemented with respect to the example apparatus, the example systems, and/or the example articles, and *vice versa.*

WHAT IS CLAIMED:

1. A computer-implemented method for video coding, comprising:

obtaining frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both; and

determining a quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of at least one current frame, and wherein each frame is given a rank associated with the level the frame is on.

2. The method of claim 1 comprising providing, at least initially, a smaller quantization parameter for the frames the closer the level of the frame is to the base level.

3. The method of claim 1 comprising providing, at least initially, the base level with the smallest quantization parameter for the frames relative to all the other levels.

4. The method of claim 1 comprising forming an at least initial Qp of the rank (0) P-picture from a predetermined relationship between Qp and a value of average temporal complexity of available complexities of future frames per target bitrate.

5. The method of claim 1 comprising using at least the initial Qp of the P-picture rank (0) frames to form the Qp of the rank (0) non-P-picture frames.

6. The method of claim 5 comprising using a predetermined mapping of P-picture rank (0) Qp value to rank (0) non-P-picture Qp value to form the rank (0) non-P-picture Qp value for a frame.

7. The method of claim 1 comprising determining a constant initial frame Qp for each non-zero level.

8. The method of claim 7 wherein the constant frame Qp is established for all frames on a non-zero level regardless of the type of frame on the non-zero level.

9. The method of claim 1 comprising determining the Qp of other than rank (0) frames from the equation:

Qp_r = inter_q[0] + ((fpic_q[0] – inter_q[0]) + (1.0/2.0)) x r,

where inter_q[0] is the initial Qp for the rank(0) P-pictures, fpic_q is the initial Qp to be used for the rank(0) F-pictures, and r is the rank of a current frame being analyzed,

wherein an option is provided to use past reference frames, future reference frames, or both for an F-picture.

10. The method of claim 1 comprising apply limits to the Qp value derived from at least one of:

(1) for frames at a start of a sequence, limits derived from predetermined values associated with the relationship between spatio-temporal complexity and bits per pixel (bpp) based on bitrate, frame rate, and width*height of a frame,

(2) for frames other than frames at a start of a sequence, limits for the frames based, at least in part, on a history including data from previously encoded frames of the same sequence as a frame being analyzed, and wherein:

the mean QP of the frames encoded so far is based on:

(avg_pqp) the average Pqps of all the frames encoded so far,

(avg_cplx) the average spatio temporal complexity,

(act_bits) the total bits consumed for all the frames encoded so far,

(cplx) the complexity of the current frame, and

(target_bits) the total target bits for all the frames including the current frame, and

wherein the history option (2) is used when the resulting mean Qp is within a certain percentage of the average Pqp, and

(3) setting the average Pqp of the frames encoded so far as the mean Qp used to determine the Qp limits,

wherein Pqp is the P-picture rank (0) Qp.

11. The method of claim 1 comprising determining a more final frame Qp relative to the initial QP differently depending on whether the frame is at the start of a sequence, is part of a blank or static scene, or neither.

12. The method of claim 1 comprising determining a more final frame QP relative to the initial QP based, at least in part, on the rank of the frame.

13. The method of claim 1 comprising establishing a value of estimated bits for a P-picture (P_est_bits) based on, at least in part, a predetermined relationship between the initial P-picture rank (0) Qp and an I-picture to P-picture bits ratio.

14. The method of claim 1 comprising establishing a value of estimated bits for an F-picture (F_est_bits) based on a predetermined relationship between the initial P-picture rank (0) Qp and a P-picture to F-picture bits ratio, wherein an option is provided to use past reference frames, future reference frames, or both for an F-picture.

15. The method of claim 8 or 13 comprising providing the option to provide the F-picture with modified reference frames that are modified by a morphing technique or a synthesis technique.

16. The method of claim 1 wherein estimation of bits of a frame are calculated with a linear model and are different for different combinations of frame type and level of the frame, wherein the frame types including at least P-picture, F-picture, B-picture, or I-picture.

17. The method of claim 1 comprising determining a more final frame Qp to maintain buffer fullness toward a mean operating point of the buffer relative to the bit capacity of the buffer.

18. The method of claim 17 wherein the buffer is a virtual buffer at an encoder that imitates the parameters of a buffer at a decoder.

19. The method of claim 1 comprising determining a more final Qp for a frame sensitive to buffer fullness of a buffer, comprising:

estimating the bits of a frame with level-sensitive Qp models;

determining whether to apply a video buffer verified (VBV) delta associated with buffer status and applied to adjust an initial Qp toward a mean operating Qp of a buffer, and applied depending on at least one of:

whether a rate of buffer increment is above a threshold, wherein the buffer increment is a value of the bits relative to an average bit buffer level for a frame sequence, and based on the estimating of the bits of a frame, and

whether a buffer estimate of the fullness of the buffer is above a threshold wherein the buffer estimate is associated with a buffer fullness value and the buffer increment value;

calculating a complexity delta associated with a weighting factor corresponding to complexity of a frame and associated with the buffer increment; and

determining the more final Qp based, at least in part, on the initial Qp adjusted by the VBV delta if the threshold is surpassed, and the complexity delta.

20. The method of claim 1 wherein the initial Qp is a rank (0) P-picture Qp.

21. The method of claim 1 comprising forming a more final Qp for a current frame depending on, at least in part, a pre-analyzing a number of lookahead frames that are future frames relative to the current frame, and wherein the pre-analyzing comprises determining the complexity and motion characteristics of the lookahead frames without fully encoding the lookahead frames.

22. The method of claim 21 comprising determining a more final Qp based, at least in part, on adjusting a more initial Qp with a delta value including determining a delta value with:

(1) a complexity delta associated with a complexity of a sequence having the current frame being analyzed and including the lookahead frames, and formed by an adaptive quantization comprising at least one of:

key frame boosting by shifting bits by changing the total bit amount of a frame to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference,

temporal complexity masking comprising shuffling bits by changing the total bit amount of a frame, from more complex to less complex frames, and

delta Qp factor smoothing of delta values of successive frames, and

(2) a video buffer verification (VBV) delta associated with the fullness of the buffer.

23. The method of claim 1 comprising:

performing a first pass analysis that encodes frames of a sequence by using the initial Qp values to obtain the bits used and complexity of the frames analyzed; and

performing a second pass analysis to determine a more final frame Qp for the frames of the sequence by adjusting the initial Qp by a delta value to maintain a QP close to a mean operating point QP corresponding to a mean operating point of a buffer holding the frames of the sequence, wherein the delta value is associated with the fullness status of a buffer and the complexity of the frame sequence being analyzed.

24. The method of claim 1 comprising:

providing, at least initially, a smaller quantization parameter for the frames the closer the level of the frame is to the base level;

providing, at least initially, the base level with the smallest quantization parameter for the frames relative to all the other levels;

forming an at least initial Qp of the rank (0) P-picture from a predetermined relationship between Qp and a value of average temporal complexity of available complexities of future frames per target bitrate;

using at least the initial Qp of the P-picture rank (0) frames to form the Qp of the rank (0) non-P-picture frames;

using a predetermined mapping of P-picture rank (0) Qp value to rank (0) non-P-picture Qp value to form the rank (0) non-P-picture Qp value for a frame;

determining a constant initial frame Qp for each non-zero level, wherein the constant frame Qp is established for all frames on a non-zero level regardless of the type of frame on the non-zero level.

determining the Qp of other than rank (0) frames from the equation:

$$Qp_r = inter\_q[0] + ((fpic\_q[0] - inter\_q[0]) + (1.0/2.0)) \times r,$$

where inter_q[0] is the initial Qp for the rank(0) P-pictures, fpic_q is the initial Qp to be used for the rank(0) F-pictures, and r is the rank of a current frame being analyzed, wherein an option is

applying limits to the Qp value derived from at least one of:

(1) for frames at a start of a sequence, limits derived from predetermined values associated with the relationship between spatio-temporal complexity and bits per pixel (bpp) based on bitrate, frame rate, and width*height of a frame,

(2) for frames other than frames at a start of a sequence, limits for the frames based, at least in part, on a history including data from previously encoded frames of the same sequence as a frame being analyzed, and wherein:

the mean QP of the frames encoded so far is based on:

(avg_pqp) the average Pqps of all the frames encoded so far,

(avg_cplx) the average spatio temporal complexity,

(act_bits) the total bits consumed for all the frames encoded so far,

(cplx) the complexity of the current frame, and

(target_bits) the total target bits for all the frames including the current frame, and wherein the history option (2) is used when the resulting mean Qp is within a certain percentage of the average Pqp, and

(3) setting the average Pqp of the frames encoded so far as the mean Qp used to determine the Qp limits, wherein Pqp is the P-picture rank (0) Qp;

determining a more final frame Qp relative to the initial QP differently depending on whether the frame is at the start of a sequence, is part of a blank or static scene, or neither;

determining a more final frame QP relative to the initial QP based, at least in part, on the rank of the frame;

establishing a value of estimated bits for a P-picture (P_est_bits) based on, at least in part, a predetermined relationship between the initial P-picture rank (0) Qp and an I-picture to P-picture bits ratio;

establishing a value of estimated bits for an F-picture (F_est_bits) based on a predetermined relationship between the initial P-picture rank (0) Qp and a P-picture to F-picture bits ratio, wherein an option is provided to use past reference frames, future reference frames, or both for an F-picture, and including providing the option to provide the F-picture with modified reference frames that are modified by a morphing technique or a synthesis technique,

wherein estimation of bits of a frame are calculated with a linear model and are different for different combinations of frame type and level of the frame, wherein the frame types including at least P-picture, F-picture, B-picture, or I-picture;

determining a more final frame Qp to maintain buffer fullness toward a mean operating point of the buffer relative to the bit capacity of the buffer, wherein the buffer is a virtual buffer at an encoder that imitates the parameters of a buffer at a decoder;

determining a more final Qp for a frame sensitive to buffer fullness of a buffer, comprising:

estimating the bits of a frame with level-sensitive Qp models;

determining whether to apply a video buffer verified (VBV) delta associated with buffer status and applied to adjust an initial Qp toward a mean operating Qp of a buffer, and applied depending on at least one of:

whether a rate of buffer increment is above a threshold, wherein the buffer increment is a value of the bits relative to an average bit buffer level for a frame sequence, and based on the estimating of the bits of a frame, and

whether a buffer estimate of the fullness of the buffer is above a threshold wherein the buffer estimate is associated with a buffer fullness value and the buffer increment value;

calculating a complexity delta associated with a weighting factor corresponding to complexity of a frame and associated with the buffer increment; and

determining the more final Qp based, at least in part, on the initial Qp adjusted by the VBV delta if the threshold is surpassed, and the complexity delta;

wherein the initial Qp is a rank (0) P-picture Qp; and

at least one of (A) and (B) comprising:

(A) performing a single-pass analysis with lookahead comprising:

forming a more final Qp for a current frame depending on, at least in part, a pre-analyzing a number of lookahead frames that are future frames relative to the current frame, and wherein the pre-analyzing comprises determining the complexity and motion characteristics of the lookahead frames without fully encoding the lookahead frames;

determining a more final Qp based, at least in part, on adjusting a more initial Qp with a delta value including determining a delta value with:

(1) a complexity delta associated with a complexity of a sequence having the current frame being analyzed and including the lookahead frames, and formed by an adaptive quantization comprising at least one of:

key frame boosting by shifting bits by changing the total bit amount of a frame to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference,

temporal complexity masking comprising shuffling bits by changing the total bit amount of a frame, from more complex to less complex frames, and

delta Qp factor smoothing of delta values of successive frames, and

(2) a video buffer verification (VBV) delta associated with the fullness of the buffer; and

(B) performing a two-pass analysis comprising:

performing a first pass analysis that encodes frames of a sequence by using the initial Qp values to obtain the bits used and complexity of the frames analyzed; and

performing a second pass analysis to determine a more final frame Qp for the frames of the sequence by adjusting the initial Qp by a delta value to maintain a QP close to a mean operating point QP corresponding to a mean operating point of a buffer holding the frames of the sequence, wherein the delta value is associated with the fullness status of a buffer and the complexity of the frame sequence being analyzed.

25. A coder comprising:

an image buffer; and

a graphics processing unit configured to:

obtain frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the

order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both, and

determine a quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of at least one current frame, and wherein each frame is given a rank associated with the level the frame is on.

26. The coder of claim 25 being further configured to:

provide, at least initially, a smaller quantization parameter for the frames the closer the level of the frame is to the base level;

provide, at least initially, the base level with the smallest quantization parameter for the frames relative to all the other levels;

form an at least initial Qp of the rank (0) P-picture from a predetermined relationship between Qp and a value of average temporal complexity of available complexities of future frames per target bitrate;

use at least the initial Qp of the P-picture rank (0) frames to form the Qp of the rank (0) non-P-picture frames;

use a predetermined mapping of P-picture rank (0) Qp value to rank (0) non-P-picture Qp value to form the rank (0) non-P-picture Qp value for a frame;

determine a constant initial frame Qp for each non-zero level, wherein the constant frame Qp is established for all frames on a non-zero level regardless of the type of frame on the non-zero level.

determine the Qp of other than rank (0) frames from the equation:

$$Qp_r = inter\_q[0] + ((fpic\_q[0] - inter\_q[0]) + (1.0/2.0)) \times r,$$

where inter_q[0] is the initial Qp for the rank(0) P-pictures, fpic_q is the initial Qp to be used for the rank(0) F-pictures, and r is the rank of a current frame being analyzed, wherein an option is

apply limits to the Qp value derived from at least one of:

(1) for frames at a start of a sequence, limits derived from predetermined values associated with the relationship between spatio-temporal complexity and bits per pixel (bpp) based on bitrate, frame rate, and width*height of a frame,

(2) for frames other than frames at a start of a sequence, limits for the frames based, at least in part, on a history including data from previously encoded frames of the same sequence as a frame being analyzed, and wherein:

the mean QP of the frames encoded so far is based on:

(avg_pqp) the average Pqps of all the frames encoded so far,

(avg_cplx) the average spatio temporal complexity,

(act_bits) the total bits consumed for all the frames encoded so far,

(cplx) the complexity of the current frame, and

(target_bits) the total target bits for all the frames including the current frame, and wherein the history option (2) is used when the resulting mean Qp is within a certain percentage of the average Pqp, and

(3) set the average Pqp of the frames encoded so far as the mean Qp used to determine the Qp limits, wherein Pqp is the P-picture rank (0) Qp;

determine a more final frame Qp relative to the initial QP differently depending on whether the frame is at the start of a sequence, is part of a blank or static scene, or neither;

determine a more final frame QP relative to the initial QP based, at least in part, on the rank of the frame;

establish a value of estimated bits for a P-picture (P_est_bits) based on, at least in part, a predetermined relationship between the initial P-picture rank (0) Qp and an I-picture to P-picture bits ratio;

establish a value of estimated bits for an F-picture (F_est_bits) based on a predetermined relationship between the initial P-picture rank (0) Qp and a P-picture to F-picture bits ratio, wherein an option is provided to use past reference frames, future reference frames, or both for

an F-picture, and including providing the option to provide the F-picture with modified reference frames that are modified by a morphing technique or a synthesis technique,

wherein estimation of bits of a frame are calculated with a linear model and are different for different combinations of frame type and level of the frame, wherein the frame types including at least P-picture, F-picture, B-picture, or I-picture;

determine a more final frame Qp to maintain buffer fullness toward a mean operating point of the buffer relative to the bit capacity of the buffer, wherein the buffer is a virtual buffer at an encoder that imitates the parameters of a buffer at a decoder;

determine a more final Qp for a frame sensitive to buffer fullness of a buffer, comprising:

estimate the bits of a frame with level-sensitive Qp models;

determine whether to apply a video buffer verified (VBV) delta associated with buffer status and applied to adjust an initial Qp toward a mean operating Qp of a buffer, and applied depending on at least one of:

whether a rate of buffer increment is above a threshold, wherein the buffer increment is a value of the bits relative to an average bit buffer level for a frame sequence, and based on the estimating of the bits of a frame, and

whether a buffer estimate of the fullness of the buffer is above a threshold wherein the buffer estimate is associated with a buffer fullness value and the buffer increment value;

calculate a complexity delta associated with a weighting factor corresponding to complexity of a frame and associated with the buffer increment; and

determine the more final Qp based, at least in part, on the initial Qp adjusted by the VBV delta if the threshold is surpassed, and the complexity delta;

wherein the initial Qp is a rank (0) P-picture Qp; and

at least one of (A) and (B) comprising:

(A) perform a single-pass analysis with lookahead comprising:

forming a more final Qp for a current frame depending on, at least in part, a pre-analyzing a number of lookahead frames that are future frames relative to the current frame, and wherein the pre-analyzing comprises determining the complexity and motion characteristics of the lookahead frames without fully encoding the lookahead frames;

determine a more final Qp based, at least in part, on adjusting a more initial Qp with a delta value including determining a delta value with:

(1) a complexity delta associated with a complexity of a sequence having the current frame being analyzed and including the lookahead frames, and formed by an adaptive quantization comprising at least one of:

key frame boosting by shifting bits by changing the total bit amount of a frame to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference,

temporal complexity masking comprising shuffling bits by changing the total bit amount of a frame, from more complex to less complex frames, and

delta Qp factor smoothing of delta values of successive frames, and

(2) a video buffer verification (VBV) delta associated with the fullness of the buffer; and

(B) perform a two-pass analysis comprising:

perform a first pass analysis that encodes frames of a sequence by using the initial Qp values to obtain the bits used and complexity of the frames analyzed; and

perform a second pass analysis to determine a more final frame Qp for the frames of the sequence by adjusting the initial Qp by a delta value to maintain a QP close to a mean operating point QP corresponding to a mean operating point of a buffer holding the frames of the sequence, wherein the delta value is associated with the fullness status of a buffer and the complexity of the frame sequence being analyzed.

27. A computer-implemented method for one-pass rate control with lookahead for video coding, comprising:

obtaining frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both;

determining an initial quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of the current frame, and wherein each frame is given a rank associated with the level the frame is on;

pre-analyzing a number of lookahead frames that are future frames relative to a current frame and analyzed to determine the complexity or motion characteristics or both of the lookahead frames without fully encoding the lookahead frames; and

determining a more final Qp by using the results of the analysis of the lookahead frames to adjust the initial Qp of the current frame.

28. The method of claim 27 comprising determining a the more final Qp based, at least in part, on determining a buffer status of a buffer that imitates the behavior of a buffer at a decoder, wherein the buffer includes space for the lookahead future frames.

29. The method of claim 28 wherein determining a buffer status comprises modulating a current Qp for frames to be close to a target mean operating point QP that corresponds to the mean operating point of the buffer.

30. The method of claim 28 wherein determining a more final Qp comprises adjusting an initial Qp by a delta value having at least part formed of a video buffer verification delta associated with the fullness status of the buffer.

31. The method of claim 28 wherein determining a more final Qp comprises adjusting an initial Qp by a delta value having at least part formed of a complexity delta associated with the complexity of the frame sequence of the current frame being analyzed and including the lookahead frames .

32. The method of claim 31 wherein the complexity delta is formed by performing adaptive quantization comprising temporal complexity masking that shuffles bits from less complex to more complex frames and by using an average complexity of the frame sequence that includes the complexities of the lookahead frames.

33. The method of claim 31 wherein the complexity data is formed by performing adaptive quantization comprising key frame boosting by shifting bits in frame sequences of low complexity to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference, wherein the frame sequence is determined to be low complexity, at least in part, by using the complexities of the lookahead frames.

34. The method of claim 31 wherein a resulting QP after a VBV delta is applied and associated with a fullness status of a buffer, adjusteing the resulting QP by applying the complexity delta.

35. The method of claim 31 wherein determining a more final Qp comprises adjusting an initial Qp by performing adaptive quantization that forms a delta to adjust the initial QP, and comprising forming the delta, at least in part, by QP factor smoothing of VBV and complexity delta values of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames.

36. The method of claim 27 comprising determining a more final Qp based, at least in part, on determining a buffer status of a buffer that imitates the behavior of a buffer at a decoder, wherein the buffer includes space for the lookahead future frames,

wherein determining a buffer status comprises modulating a current Qp for frames to be close to a target mean operating point QP that corresponds to the mean operating point of the buffer,

wherein determining a more final Qp comprises adjusting an initial Qp by a delta value having at least part formed of at least one of:

a video buffer verification delta associated with the fullness status of the buffer, and

a complexity delta associated with the complexity of the frame sequence of the current frame being analyzed and including the lookahead frames, wherein the complexity delta is formed by performing adaptive quantization comprising temporal complexity masking that shuffles bits from less complex to more complex frames and by using an average complexity of the frame sequence that includes the complexities of the lookahead frames,

wherein the complexity data is formed by performing adaptive quantization comprising key frame boosting by shifting bits in frame sequences of low complexity to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference, wherein the frame sequence is determined to be low complexity, at least in part, by using the complexities of the lookahead frames,

wherein a resulting QP after a VBV delta is applied and associated with a fullness status of a buffer, adjusting the resulting QP by applying the complexity delta,

wherein determining a more final Qp comprises adjusting an initial Qp by performing adaptive quantization that forms a delta to adjust the initial QP, and comprising forming the delta, at least in part, by QP factor smoothing of VBV and complexity delta values of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames.

37. A coder, comprising:

an image buffer; and

a graphics processing unit configured to:

obtain frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both;

determine an initial quantization parameter (Qp) for the frames depending at least on the level of the hierarchy of the current frame, and wherein each frame is given a rank associated with the level the frame is on;

pre-analyze a number of lookahead frames that are future frames relative to a current frame and analyzed to determine the complexity or motion characteristics or both of the lookahead frames without fully encoding the lookahead frames; and

determine a more final Qp by using the results of the analysis of the lookahead frames to adjust the initial Qp of the current frame.

38. The coder of claim 37 wherein the coder is further configured to:

determine a the more final Qp based, at least in part, on determining a buffer status of a buffer that imitates the behavior of a buffer at a decoder, wherein the buffer includes space for the lookahead future frames,

wherein determining a buffer status comprises modulating a current Qp for frames to be close to a target mean operating point QP that corresponds to the mean operating point of the buffer,

wherein determining a more final Qp comprises adjusting an initial Qp by a delta value having at least part formed of at least one of:

a video buffer verification delta associated with the fullness status of the buffer, and

a complexity delta associated with the complexity of the frame sequence of the current frame being analyzed and including the lookahead frames, wherein the complexity delta is formed by performing adaptive quantization comprising temporal complexity masking that shuffles bits from less complex to more complex frames and by using an average complexity of the frame sequence that includes the complexities of the lookahead frames,

wherein the complexity data is formed by performing adaptive quantization comprising key frame boosting by shifting bits in frame sequences of low complexity to an I-picture and from other than I-pictures in the same sequence and that use the I-picture as a reference, wherein the frame sequence is determined to be low complexity, at least in part, by using the complexities of the lookahead frames,

wherein a resulting QP after a VBV delta is applied and associated with a fullness status of a buffer, adjusting the resulting QP by applying the complexity delta,

wherein determining a more final Qp comprises adjusting an initial Qp by performing adaptive quantization that forms a delta to adjust the initial QP, and comprising forming the delta, at least in part, by QP factor smoothing of VBV and complexity delta values of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames.

39. A computer-implemented method for two-pass rate control for video coding, comprising:

obtaining frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the

order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both;

performing a first pass analysis with initial default quantization parameter (Qp) for each frame analyzed, wherein the initial default Qps are established at least partly based on the level of the hierarchy of the current frame, and wherein each frame is given a rank associated with the level the frame is on, and performing a first pass analysis comprising encoding the frames sufficiently to determine a complexity and bits for the frames analyzed; and

performing a second pass analysis comprising determining a more final quantization parameter (Qp) for individual frames and for coding comprising:

performing adaptive quantization to enhance the quality of one or more frames comprising shifting bits by raising the total amount of bits for at least one frame while reducing the amount of bits for at least one other frame in the same sequence; and

determining a target Qp for a frame to be used for coding and target bits for a frame for updating statistics of the first pass analysis.

40. The method of claim 39 wherein the first pass analysis encodes all of the frames in the sequence.

41. The method of claim 39 wherein the initial default rank (0) P-picture is formed by using a predetermined relationship between complexity, bitrate, and Qp.

42. The method of claim 39 comprising scaling the frames to match a target bits per frame.

43. The method of claim 39 wherein performing adaptive quantization comprises key frame boosting comprising shifting bits to an I-frame and from the plurality of frames that use the I-frame as a reference.

44. The method of claim 39 wherein performing adaptive quantization comprises temporal complexity masking including shuffling from more complex frames from less complex frames within a sequence, and depending on the same type and rank frame combinations

45. The method of claim 39 wherein the scaling and adaptive quantization result in a target bit frame size, and the method comprising determining estimated final Qp for coding of a frame depending on the bits and Qp of the frame determined in the first pass analysis and the target bit frame size (tgt_size).

46. The method of claim 39 wherein performing adaptive quantization forms a delta Qp factor, and comprises QP factor smoothing of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames.

47. The method of claim 39 comprising determining a more final Qp by using an iteration that factors in the frame rank and type of the frame being analyzed,

48. The method of claim 47 wherein the iteration also factors in the delta Qp factor of adaptive quantization.

49. The method of claim 47 wherein the iteration also factors in the bits per frame and Qp from the first pass analysis.

50. The method of claim 47 wherein the iteration comprises:

setting a test Qp value based on a step value;

calculating an expected total bits for a chosen frame wherein the expected bits is based on, at least in part, frame rank and type of the frame being analyzed, delta Qp factor of adaptive quantization, and bits per frame and Qp from the first pass analysis;

comparing the expected total bits to a target bits value as a threshold;

changing the test Qp value differently depending on whether the threshold is surpassed; and

restarting the process with setting a test Qp for a finite number of times.

51. The method of claim 39 wherein the first pass analysis encodes all of the frames in the sequence,

wherein the initial default rank (0) P-picture is formed by using a predetermined relationship between complexity, bitrate, and Qp,

the method comprising scaling the frames to match a target bits per frame,

wherein performing adaptive quantization comprises key frame boosting comprising shifting bits to an I-frame and from the plurality of frames that use the I-frame as a reference,

wherein performing adaptive quantization comprises temporal complexity masking including shuffling from more complex frames from less complex frames within a sequence, and depending on the same type and rank frame combinations,

wherein the scaling and adaptive quantization result in a target bit frame size, and the method comprising determining estimated final Qp for coding of a frame depending on the bits and Qp of the frame determined in the first pass analysis and the target bit frame size (tgt_size),

wherein performing adaptive quantization forms a delta Qp factor, and comprises QP factor smoothing of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames,

the method comprising determining a more final Qp by using an iteration that factors in the frame rank and type of the frame being analyzed, wherein the iteration also factors in the delta Qp factor of adaptive quantization, and wherein the iteration also factors in the bits per frame and Qp from the first pass analysis, wherein the iteration comprises:

setting a test Qp value based on a step value;

calculating an expected total bits for a chosen frame wherein the expected bits is based on, at least in part, frame rank and type of the frame being analyzed, delta Qp factor of adaptive quantization, and bits per frame and Qp from the first pass analysis;

comparing the expected total bits to a target bits value as a threshold;

changing the test Qp value differently depending on whether the threshold is surpassed; and

restarting the process with setting a test Qp for a finite number of times.

52. A coder, comprising:

an image buffer; and

a graphics processing unit configured to:

obtain frames of pixel data in an input video order and associated with a multi-level hierarchy comprising a base level with at least I-pictures or P-pictures or both that are used as reference frames, at least one intermediate level with pictures that use frames on the base level as references, and a maximum level with pictures that are not used as reference frames, and that use the frames of the other levels as references, wherein P-pictures use past frames relative to the order as references, and wherein pictures on the maximum level are provided with the option to use past reference frames, future reference frames, or both;

perform a first pass analysis with initial default quantization parameter (Qp) for each frame analyzed, wherein the initial default Qps are established at least partly based on the level of the hierarchy of the current frame, and wherein each frame is given a rank associated with the level the frame is on, and performing a first pass analysis comprising encoding the frames sufficiently to determine a complexity and bits for the frames analyzed;

perform a second pass analysis comprising determining a more final quantization parameter (Qp) for individual frames and for coding comprising:

perform adaptive quantization to enhance the quality of one or more frames comprising shifting bits by raising the total amount of bits for at least one frame while reducing the amount of bits for at least one other frame in the same sequence; and

determine a target Qp for a frame to be used for coding and target bits for a frame for updating statistics of the first pass analysis.

53. The coder of claim 52 wherein the first pass analysis encodes all of the frames in the sequence,

wherein the initial default rank (0) P-picture is formed by using a predetermined relationship between complexity, bitrate, and Qp,

the coder being configured to scale the frames to match a target bits per frame,

wherein performing adaptive quantization comprises key frame boosting comprising shifting bits to an I-frame and from the plurality of frames that use the I-frame as a reference,

wherein performing adaptive quantization comprises temporal complexity masking including shuffling from more complex frames from less complex frames within a sequence, and depending on the same type and rank frame combinations,

wherein the scaling and adaptive quantization result in a target bit frame size, and the method comprising determining estimated final Qp for coding of a frame depending on the bits and Qp of the frame determined in the first pass analysis and the target bit frame size (tgt_size),

wherein performing adaptive quantization forms a delta Qp factor, and comprises QP factor smoothing of successive frames to limit variation in QP between successive frames and depending on the type and rank combination of the frames,

the coder being configured to determine a more final Qp by using an iteration that factors in the frame rank and type of the frame being analyzed, wherein the iteration also factors in the delta Qp factor of adaptive quantization, and wherein the iteration also factors in the bits per frame and Qp from the first pass analysis, wherein the iteration comprises:

setting a test Qp value based on a step value;

calculating an expected total bits for a chosen frame wherein the expected bits is based on, at least in part, frame rank and type of the frame being analyzed, delta Qp factor of adaptive quantization, and bits per frame and Qp from the first pass analysis;

comparing the expected total bits to a target bits value as a threshold;

changing the test Qp value differently depending on whether the threshold is surpassed; and

restarting the process with setting a test Qp for a finite number of times.

54. At least one machine readable medium comprising a plurality of instructions that in response to being executed on a computing device, causes the computing device to perform the method according to any one of claims 1-24.

55. An apparatus comprising means for performing the methods according to any one of claims 1-24.

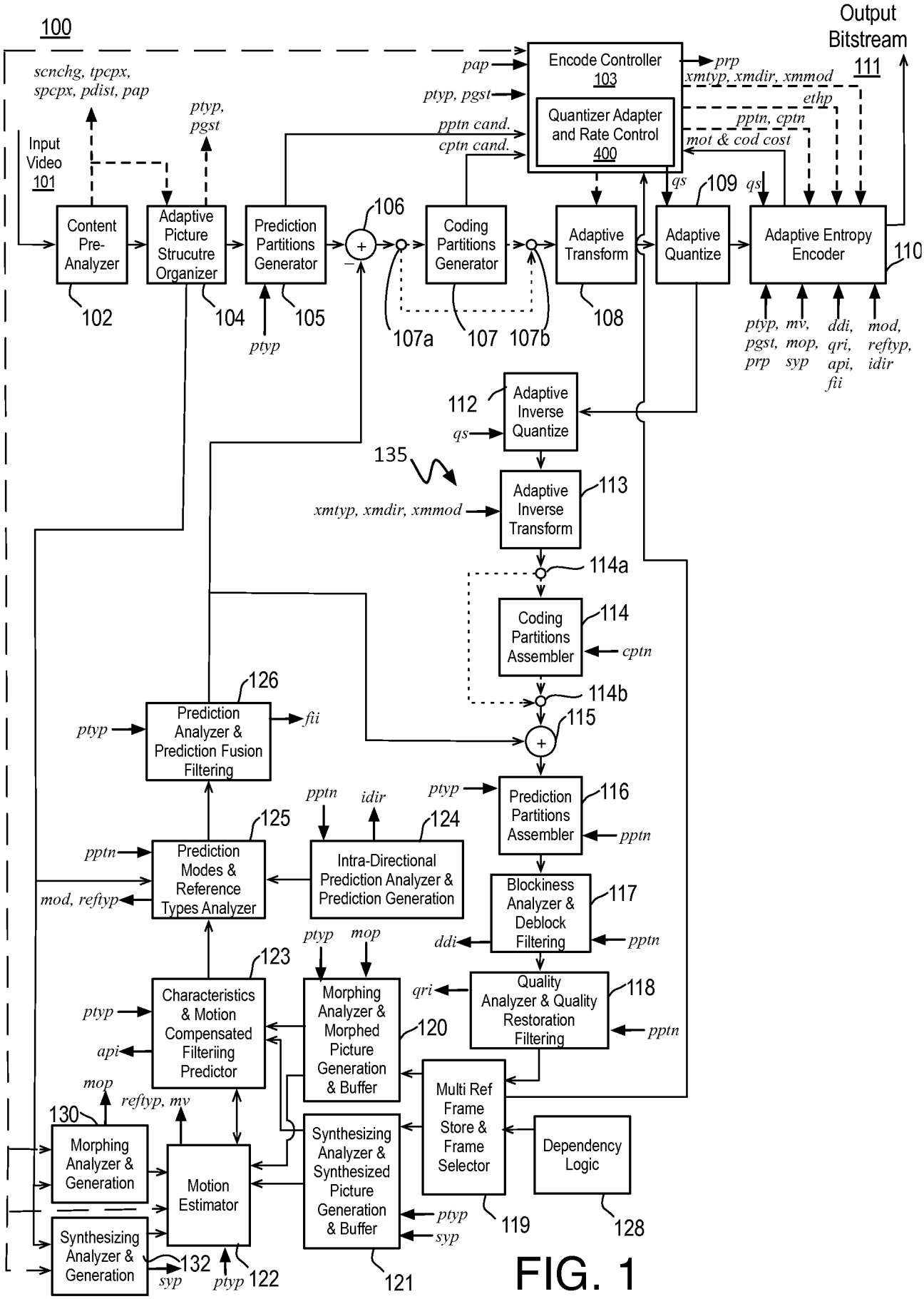

100
Output Bitstream
111
Input Video 101
scnchg, tpcpx, spcpx, pdist, pap
ptyp, pgst
pap
ptyp, pgst
pptn cand.
cptn cand.
Encode Controller 103
Quantizer Adapter and Rate Control 400
prp
xmtyp, xmdir, xmmod
ethp
pptn, cptn
mot & cod cost
qs
109
Content Pre-Analyzer
Adaptive Picture Strucutre Organizer
Prediction Partitions Generator
106
Coding Partitions Generator
Adaptive Transform
Adaptive Quantize
Adaptive Entropy Encoder
110
102
104
105
107a
107
107b
108
ptyp, pgst, prp
mv, mop, syp
ddi, qri, api, fii
mod, reftyp, idir
112
Adaptive Inverse Quantize
135
Adaptive Inverse Transform
113
xmtyp, xmdir, xmmod
114a
Coding Partitions Assembler
114
cptn
114b
115
126
Prediction Analyzer & Prediction Fusion Filtering
fii
Prediction Partitions Assembler
116
pptn
Prediction Modes & Reference Types Analyzer
125
Intra-Directional Prediction Analyzer & Prediction Generation
124
mod, reftyp
Blockiness Analyzer & Deblock Filtering
117
ddi
Quality Analyzer & Quality Restoration Filtering
118
qri
Characteristics & Motion Compensated Filteriing Predictor
123
Morphing Analyzer & Morphed Picture Generation & Buffer
120
api
mop
reftyp, mv
Multi Ref Frame Store & Frame Selector
130
Morphing Analyzer & Generation
Motion Estimator
Synthesizing Analyzer & Synthesized Picture Generation & Buffer
Dependency Logic
119
128
Synthesizing Analyzer & Generation
132
syp
122
121

FIG. 1

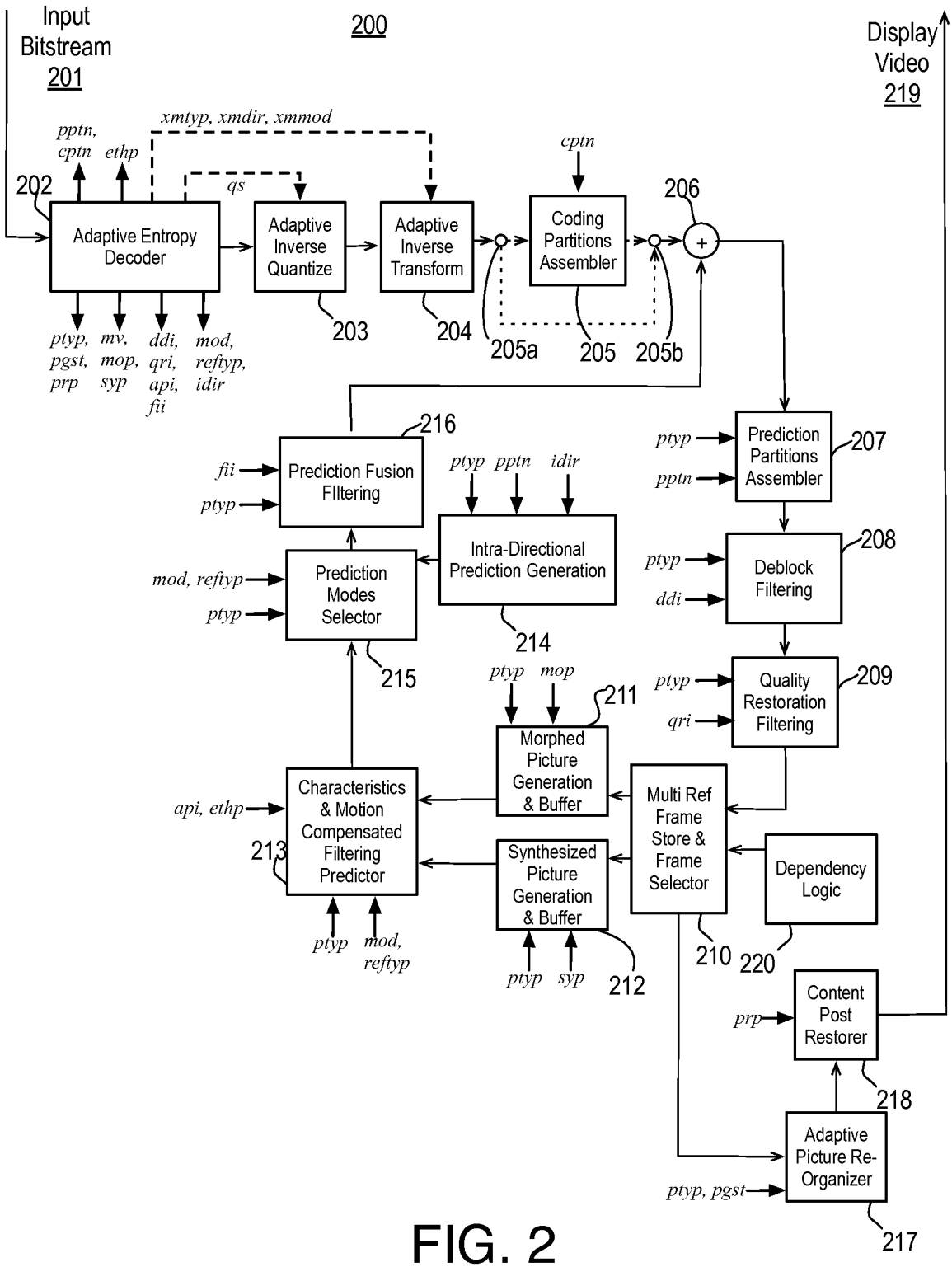

Input Bitstream 201
200
Display Video 219
xmtyp, xmdir, xmmod
pptn, cptn ethp
qs
cptn
202
206
Adaptive Entropy Decoder
Adaptive Inverse Quantize
Adaptive Inverse Transform
Coding Partitions Assembler
+
ptyp, pgst, prp
mv, mop, syp
ddi, qri, api, fii
mod, reftyp, idir
203
204
205a
205
205b
216
fii
ptyp
Prediction Fusion FIltering
ptyp pptn idir
Intra-Directional Prediction Generation
mod, reftyp
ptyp
Prediction Modes Selector
214
215
ptyp
pptn
Prediction Partitions Assembler
207
ptyp
ddi
Deblock Filtering
208
ptyp
qri
Quality Restoration Filtering
209
ptyp mop
211
Morphed Picture Generation & Buffer
api, ethp
Characteristics & Motion Compensated Filtering Predictor
213
Synthesized Picture Generation & Buffer
Multi Ref Frame Store & Frame Selector
Dependency Logic
ptyp mod, reftyp
ptyp syp
212
210
220
prp
Content Post Restorer
218
Adaptive Picture Re-Organizer
ptyp, pgst
217

FIG. 2

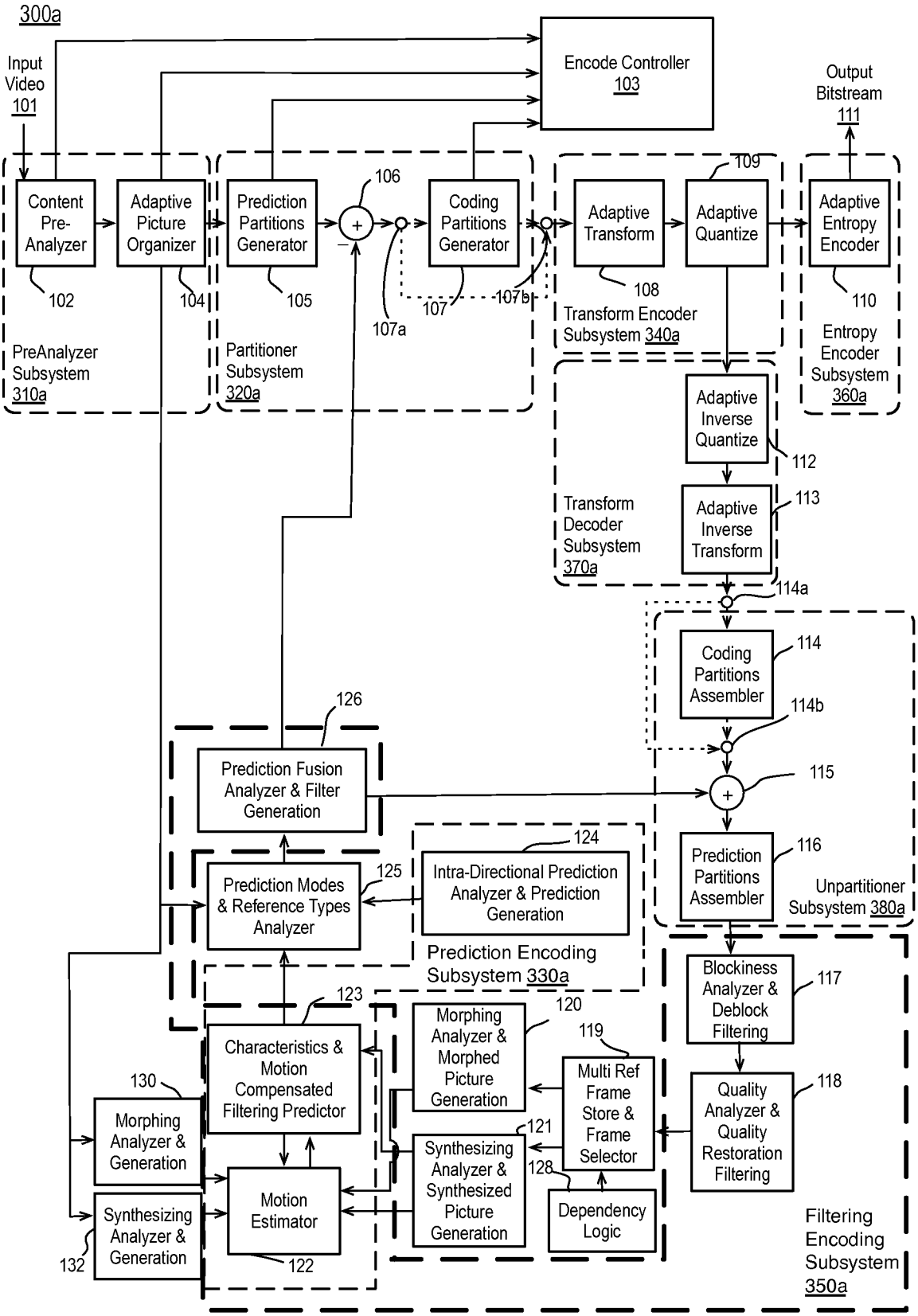

300a
Input Video 101
Encode Controller 103
Output Bitstream 111
Content Pre-Analyzer
Adaptive Picture Organizer
Prediction Partitions Generator
Coding Partitions Generator
Adaptive Transform
Adaptive Quantize
Adaptive Entropy Encoder
102
104
105
106
107
107a
107b
108
109
110
PreAnalyzer Subsystem 310a
Partitioner Subsystem 320a
Transform Encoder Subsystem 340a
Entropy Encoder Subsystem 360a
Adaptive Inverse Quantize
Adaptive Inverse Transform
112
113
Transform Decoder Subsystem 370a
114a
Coding Partitions Assembler
114
114b
115
Prediction Partitions Assembler
116
Unpartitioner Subsystem 380a
126
Prediction Fusion Analyzer & Filter Generation
124
Intra-Directional Prediction Analyzer & Prediction Generation
125
Prediction Modes & Reference Types Analyzer
Prediction Encoding Subsystem 330a
Blockiness Analyzer & Deblock Filtering
117
123
120
119
Characteristics & Motion Compensated Filtering Predictor
Morphing Analyzer & Morphed Picture Generation
Multi Ref Frame Store & Frame Selector
Quality Analyzer & Quality Restoration Filtering
118
130
121
128
Morphing Analyzer & Generation
Synthesizing Analyzer & Synthesized Picture Generation
Dependency Logic
Motion Estimator
Synthesizing Analyzer & Generation
132
122
Filtering Encoding Subsystem 350a

FIG. 3(a)

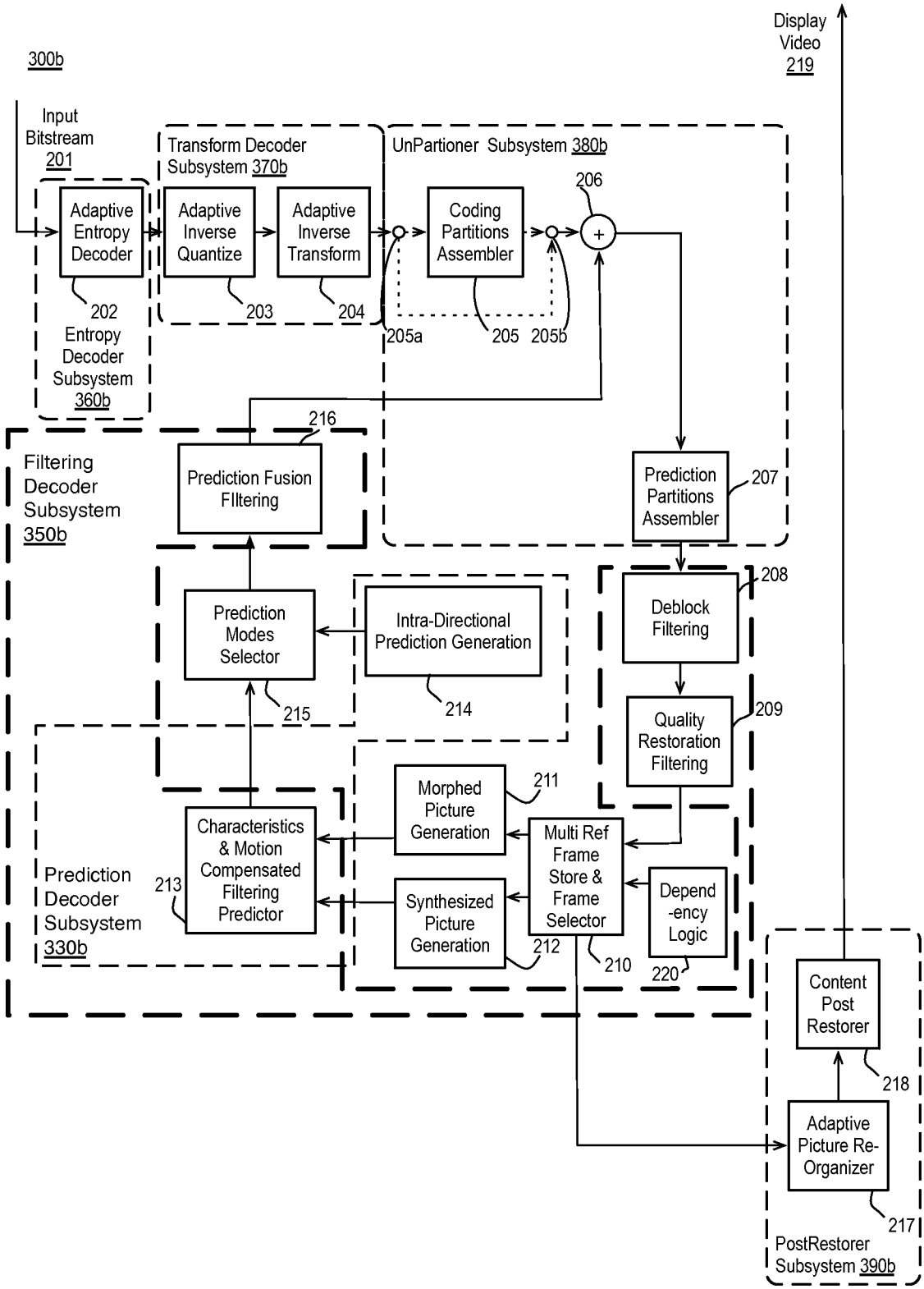

300b
Display Video 219
Input Bitstream 201
Transform Decoder Subsystem 370b
UnPartioner Subsystem 380b
Adaptive Entropy Decoder
Adaptive Inverse Quantize
Adaptive Inverse Transform
Coding Partitions Assembler
206
202 Entropy Decoder Subsystem 360b
203
204
205a
205
205b
216
Filtering Decoder Subsystem 350b
Prediction Fusion FIltering
Prediction Partitions Assembler
207
208
Deblock Filtering
Prediction Modes Selector
Intra-Directional Prediction Generation
215
214
209
Quality Restoration Filtering
211
Morphed Picture Generation
Characteristics & Motion Compensated Filtering Predictor
Multi Ref Frame Store & Frame Selector
Depend -ency Logic
Prediction Decoder Subsystem 330b
213
Synthesized Picture Generation
212
210
220
Content Post Restorer
218
Adaptive Picture Re-Organizer
217
PostRestorer Subsystem 390b

FIG. 3(b)

FIG. 4

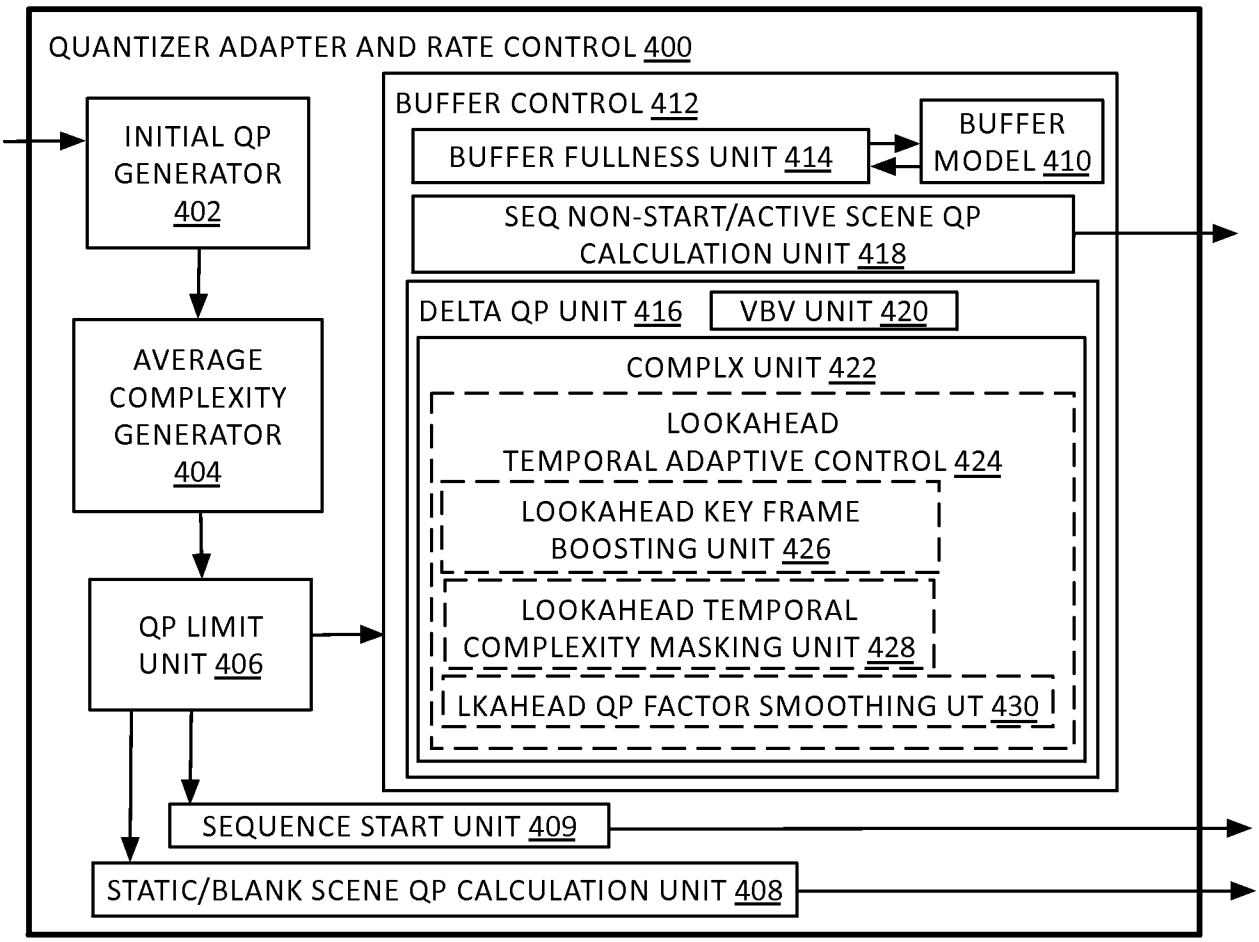

QUANTIZER ADAPTER AND RATE CONTROL 400
INITIAL QP GENERATOR 402
AVERAGE COMPLEXITY GENERATOR 404
QP LIMIT UNIT 406
BUFFER CONTROL 412
BUFFER FULLNESS UNIT 414
BUFFER MODEL 410
SEQ NON-START/ACTIVE SCENE QP CALCULATION UNIT 418
DELTA QP UNIT 416
VBV UNIT 420
COMPLX UNIT 422
LOOKAHEAD TEMPORAL ADAPTIVE CONTROL 424
LOOKAHEAD KEY FRAME BOOSTING UNIT 426
LOOKAHEAD TEMPORAL COMPLEXITY MASKING UNIT 428
LKAHEAD QP FACTOR SMOOTHING UT 430
SEQUENCE START UNIT 409
STATIC/BLANK SCENE QP CALCULATION UNIT 408

FIG. 5

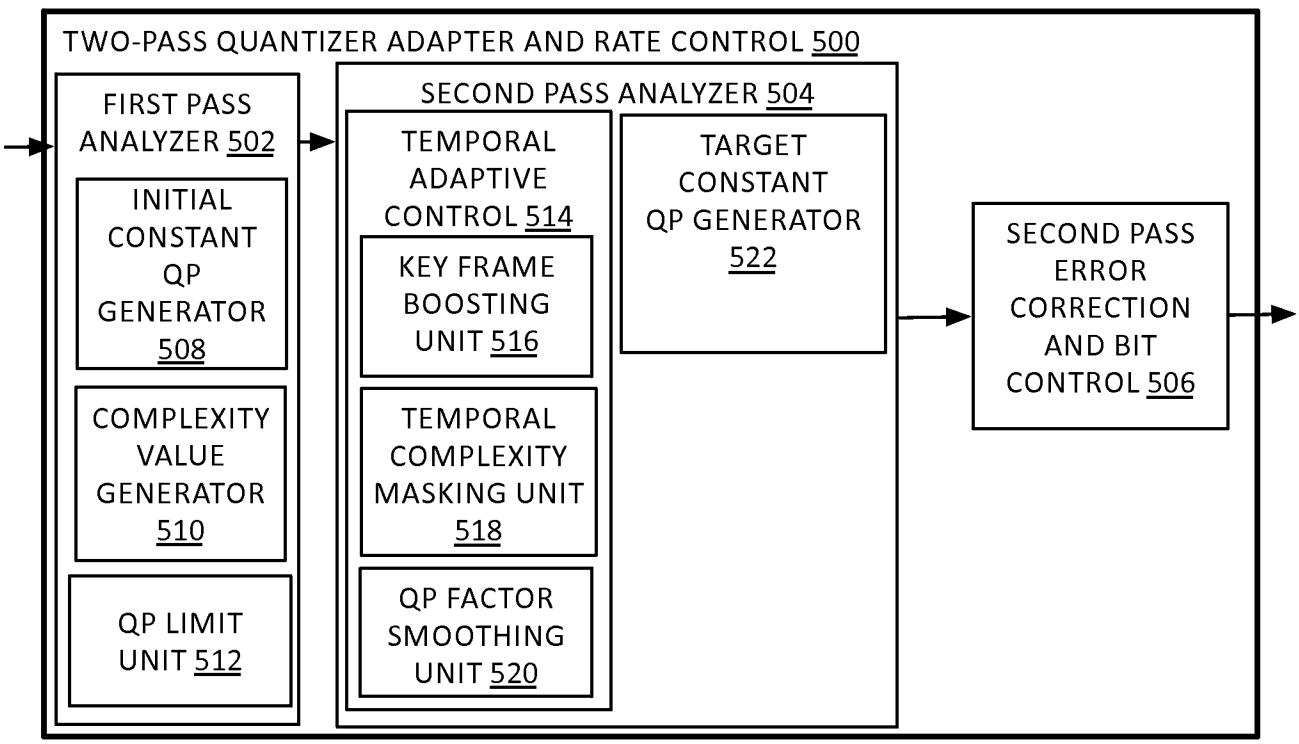

TWO-PASS QUANTIZER ADAPTER AND RATE CONTROL 500
FIRST PASS ANALYZER 502
INITIAL CONSTANT QP GENERATOR 508
COMPLEXITY VALUE GENERATOR 510
QP LIMIT UNIT 512
SECOND PASS ANALYZER 504
TEMPORAL ADAPTIVE CONTROL 514
KEY FRAME BOOSTING UNIT 516
TEMPORAL COMPLEXITY MASKING UNIT 518
QP FACTOR SMOOTHING UNIT 520
TARGET CONSTANT QP GENERATOR 522
SECOND PASS ERROR CORRECTION AND BIT CONTROL 506

FIG. 6

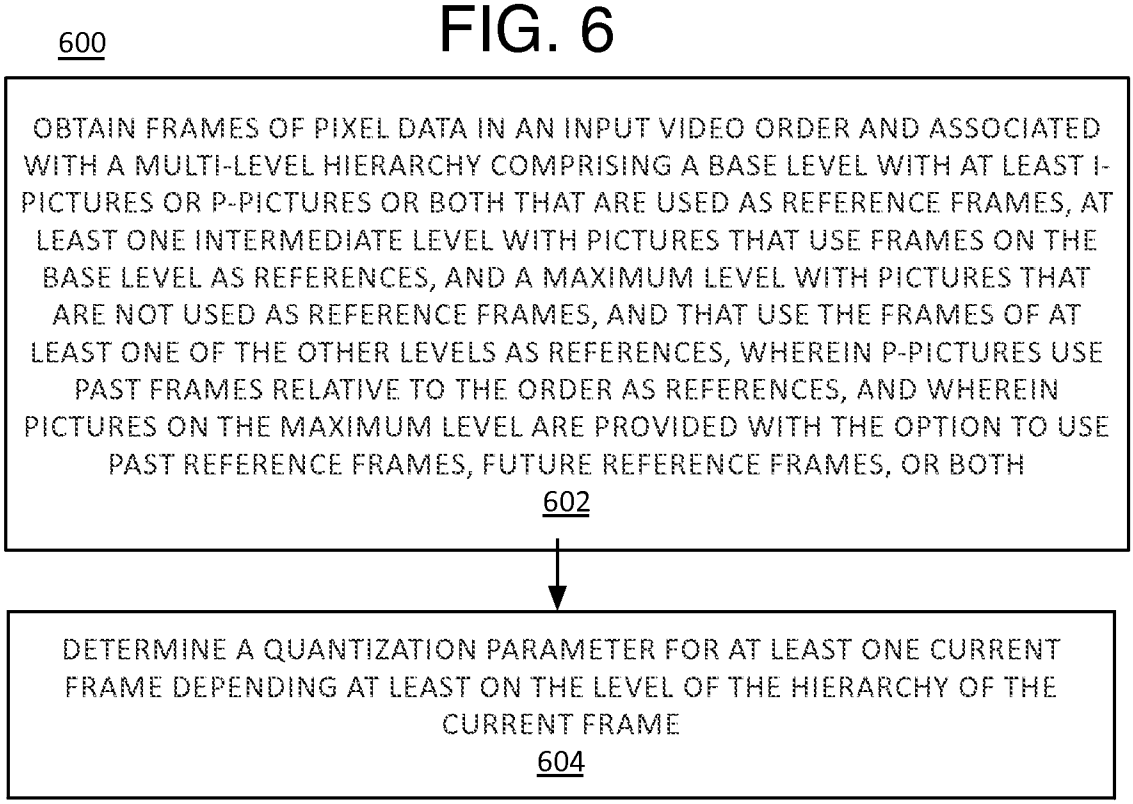

600
OBTAIN FRAMES OF PIXEL DATA IN AN INPUT VIDEO ORDER AND ASSOCIATED WITH A MULTI-LEVEL HIERARCHY COMPRISING A BASE LEVEL WITH AT LEAST I-PICTURES OR P-PICTURES OR BOTH THAT ARE USED AS REFERENCE FRAMES, AT LEAST ONE INTERMEDIATE LEVEL WITH PICTURES THAT USE FRAMES ON THE BASE LEVEL AS REFERENCES, AND A MAXIMUM LEVEL WITH PICTURES THAT ARE NOT USED AS REFERENCE FRAMES, AND THAT USE THE FRAMES OF AT LEAST ONE OF THE OTHER LEVELS AS REFERENCES, WHEREIN P-PICTURES USE PAST FRAMES RELATIVE TO THE ORDER AS REFERENCES, AND WHEREIN PICTURES ON THE MAXIMUM LEVEL ARE PROVIDED WITH THE OPTION TO USE PAST REFERENCE FRAMES, FUTURE REFERENCE FRAMES, OR BOTH
602
DETERMINE A QUANTIZATION PARAMETER FOR AT LEAST ONE CURRENT FRAME DEPENDING AT LEAST ON THE LEVEL OF THE HIERARCHY OF THE CURRENT FRAME
604

FIG. 8

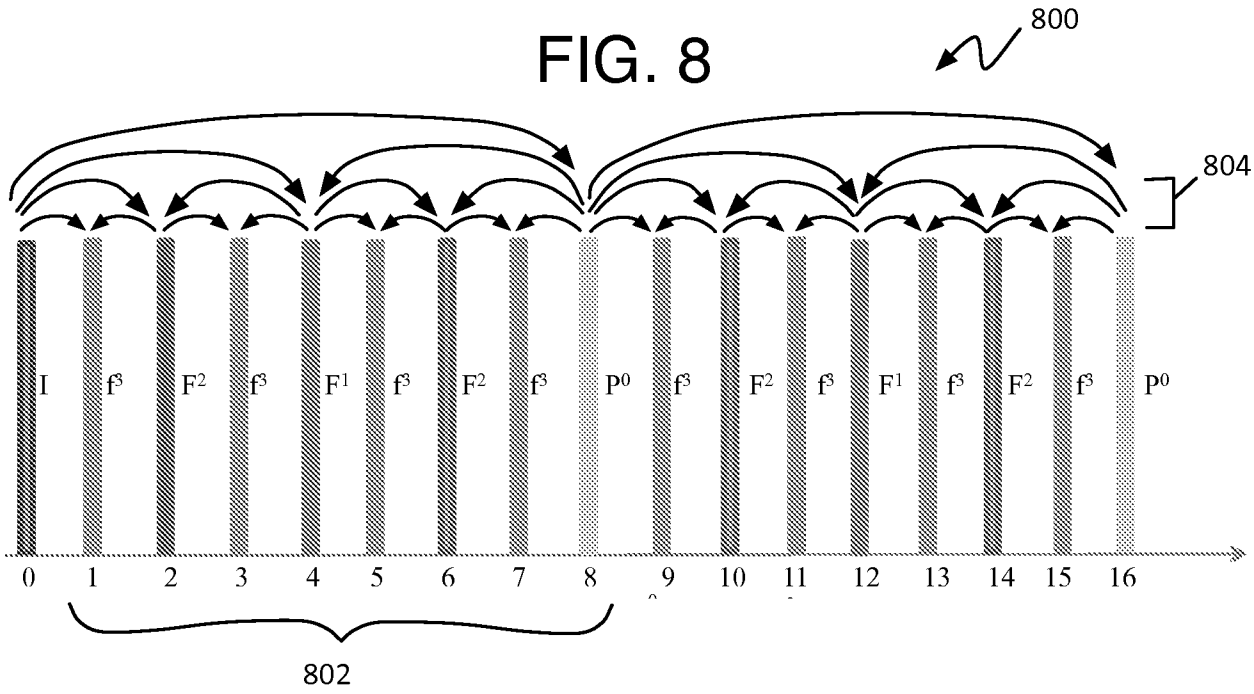

800
804
I
f3
F2
f3
F1
f3
F2
f3
P0
f3
F2
f3
F1
f3
F2
f3
P0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
802

FIG. 7

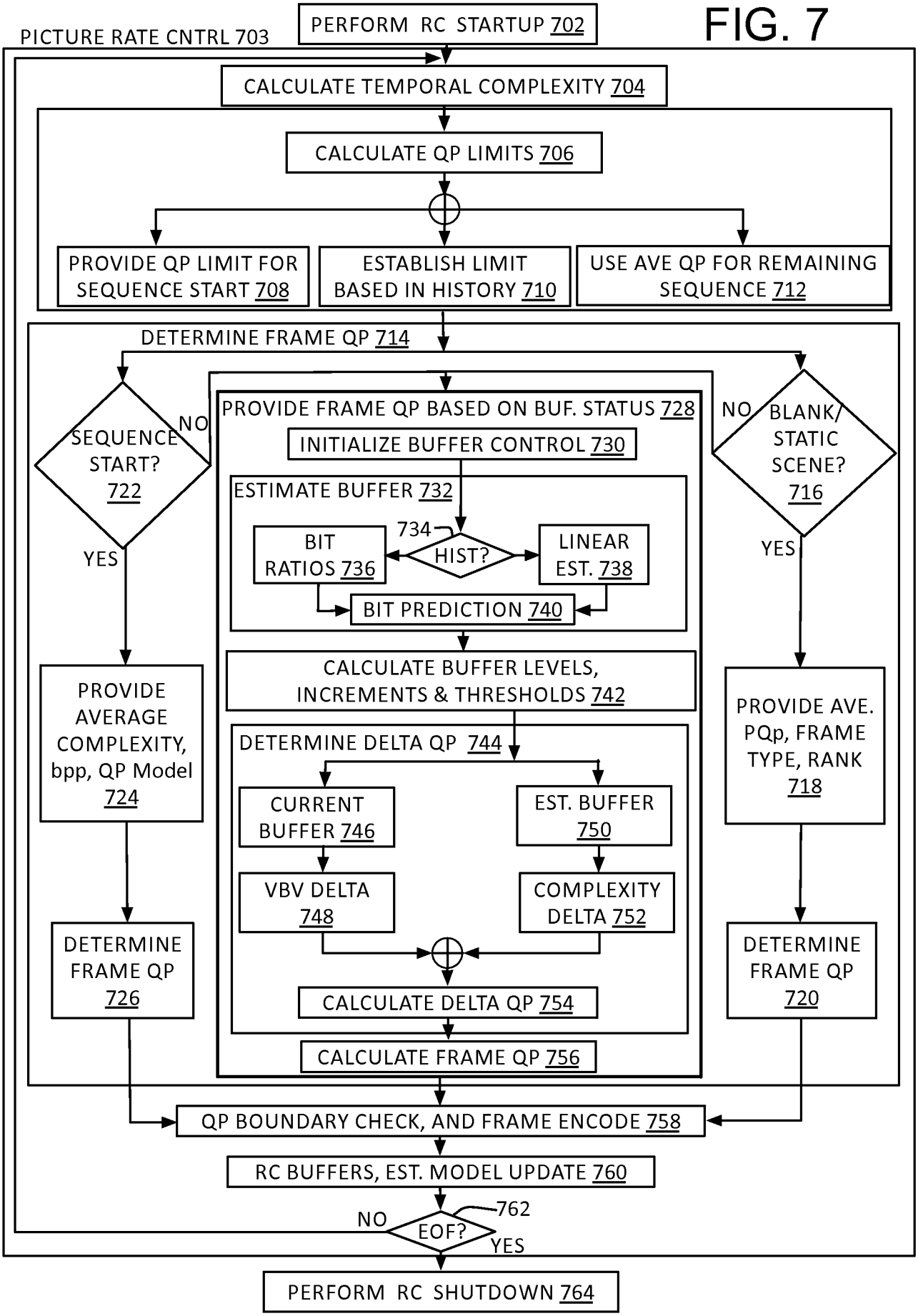

PICTURE RATE CNTRL 703
PERFORM RC STARTUP 702
CALCULATE TEMPORAL COMPLEXITY 704
CALCULATE QP LIMITS 706
PROVIDE QP LIMIT FOR SEQUENCE START 708
ESTABLISH LIMIT BASED IN HISTORY 710
USE AVE QP FOR REMAINING SEQUENCE 712
DETERMINE FRAME QP 714
SEQUENCE START? 722
NO
YES
BLANK/ STATIC SCENE? 716
NO
YES
PROVIDE FRAME QP BASED ON BUF. STATUS 728
INITIALIZE BUFFER CONTROL 730
ESTIMATE BUFFER 732
734
HIST?
BIT RATIOS 736
LINEAR EST. 738
BIT PREDICTION 740
CALCULATE BUFFER LEVELS, INCREMENTS & THRESHOLDS 742
DETERMINE DELTA QP 744
CURRENT BUFFER 746
EST. BUFFER 750
VBV DELTA 748
COMPLEXITY DELTA 752
CALCULATE DELTA QP 754
CALCULATE FRAME QP 756
PROVIDE AVERAGE COMPLEXITY, bpp, QP Model 724
PROVIDE AVE. PQp, FRAME TYPE, RANK 718
DETERMINE FRAME QP 726
DETERMINE FRAME QP 720
QP BOUNDARY CHECK, AND FRAME ENCODE 758
RC BUFFERS, EST. MODEL UPDATE 760
762
NO
EOF?
YES
PERFORM RC SHUTDOWN 764

FIG. 9

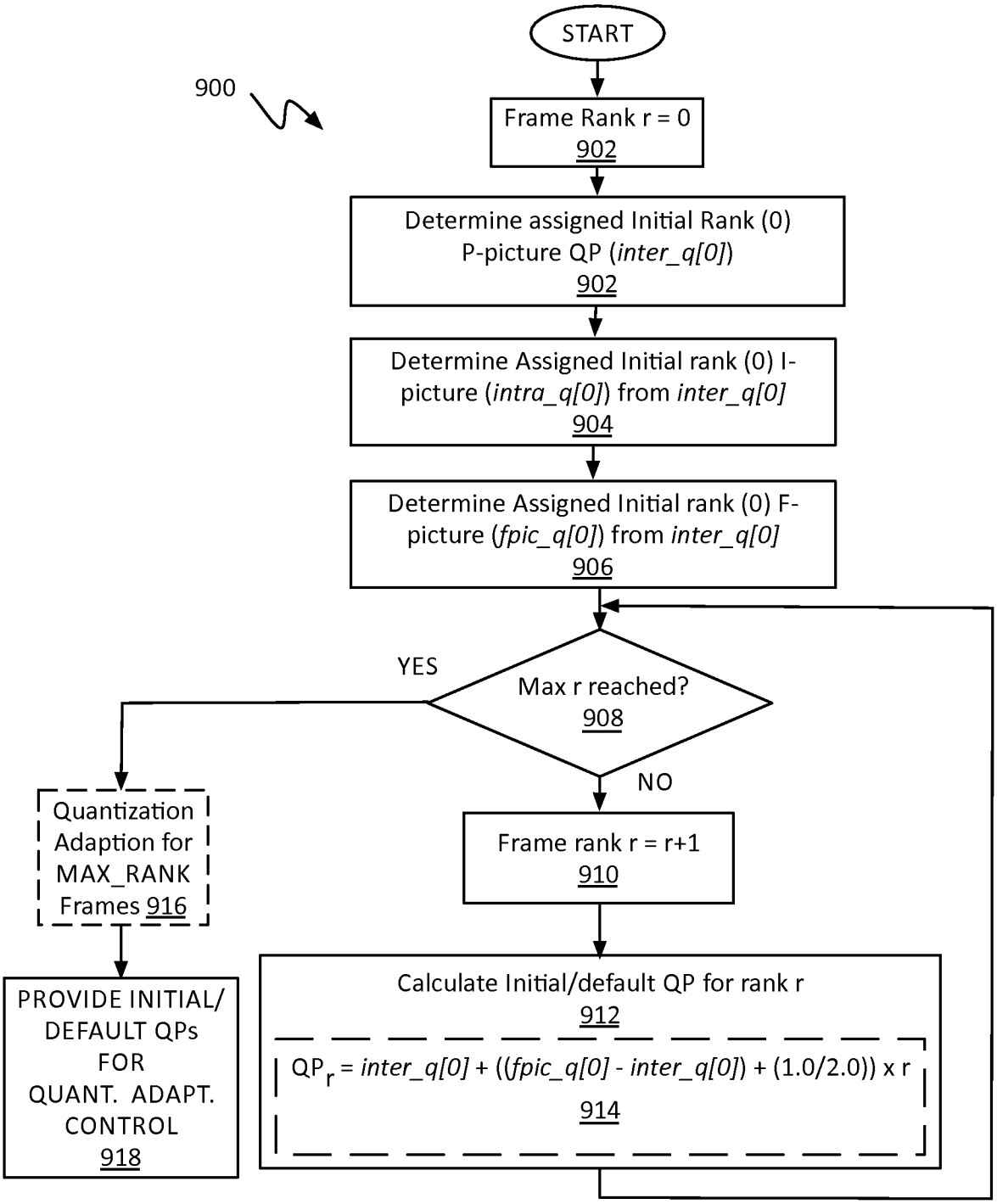

900
START
Frame Rank r = 0
902
Determine assigned Initial Rank (0) P-picture QP (inter_q[0])
902
Determine Assigned Initial rank (0) I-picture (intra_q[0]) from inter_q[0]
904
Determine Assigned Initial rank (0) F-picture (fpic_q[0]) from inter_q[0]
906
Max r reached?
908
YES
NO
Quantization Adaption for MAX_RANK Frames 916
PROVIDE INITIAL/ DEFAULT QPs FOR QUANT. ADAPT. CONTROL
918
Frame rank r = r+1
910
Calculate Initial/default QP for rank r
912
QP r = inter_q[0] + ((fpic_q[0] - inter_q[0]) + (1.0/2.0)) x r
914

FIG. 10

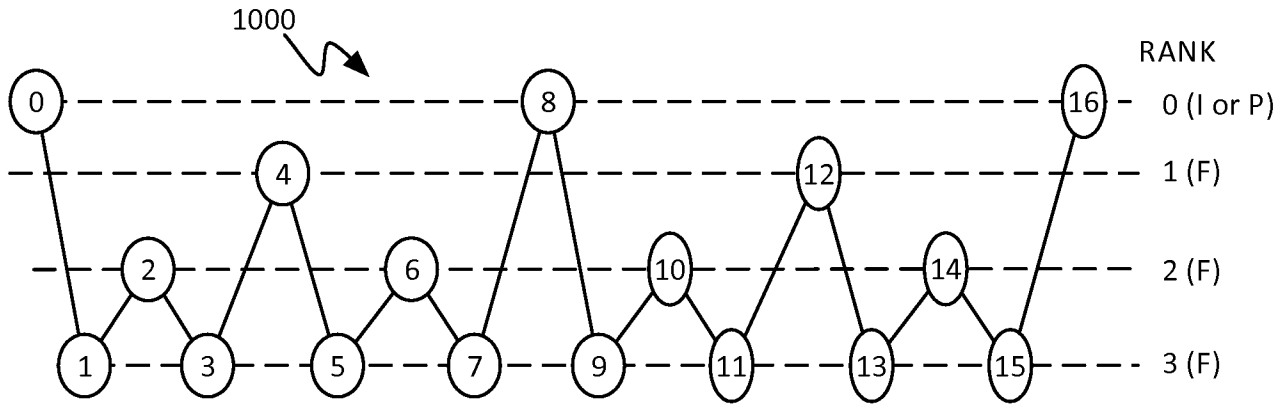


FIG. 11

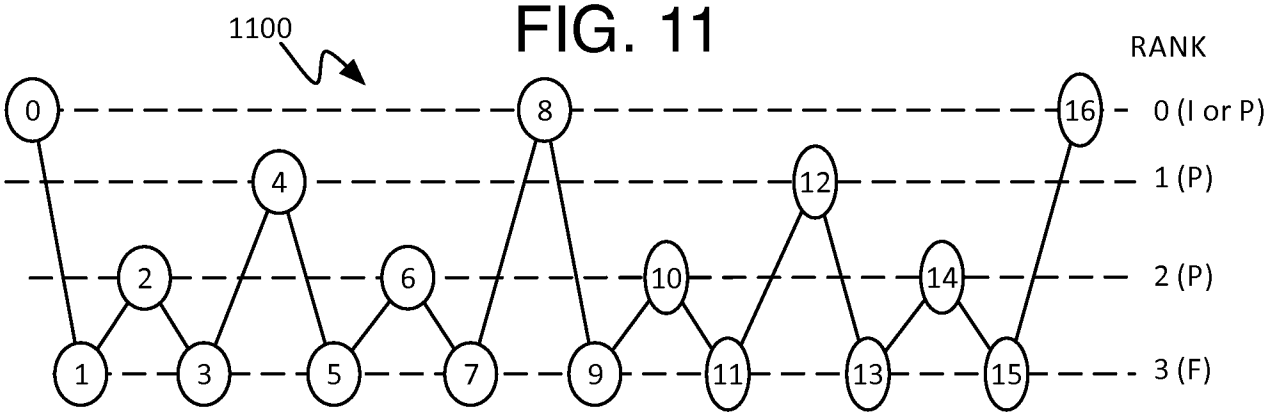


FIG. 12

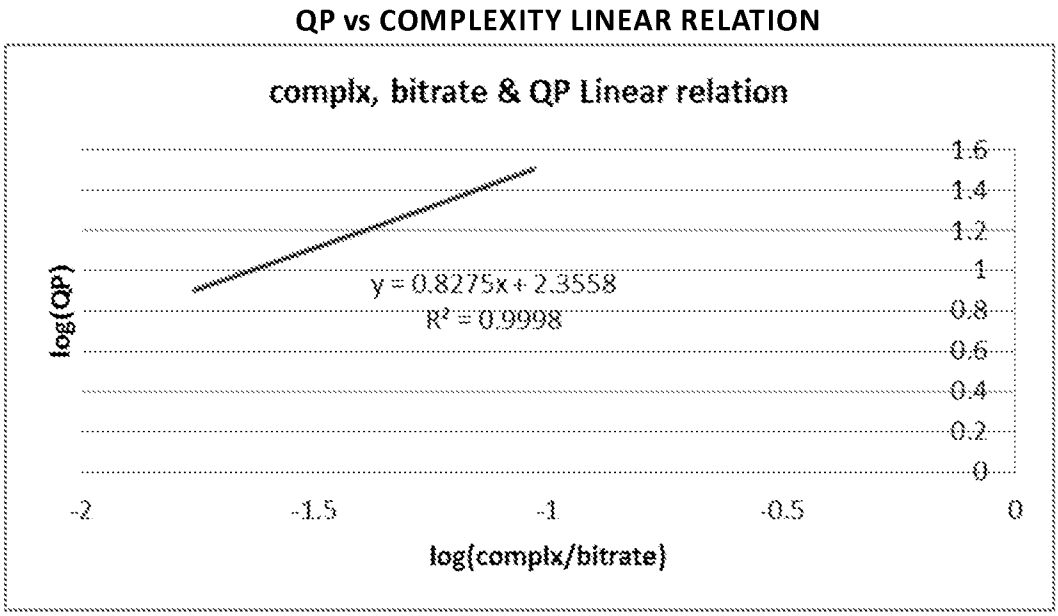


FIG. 13

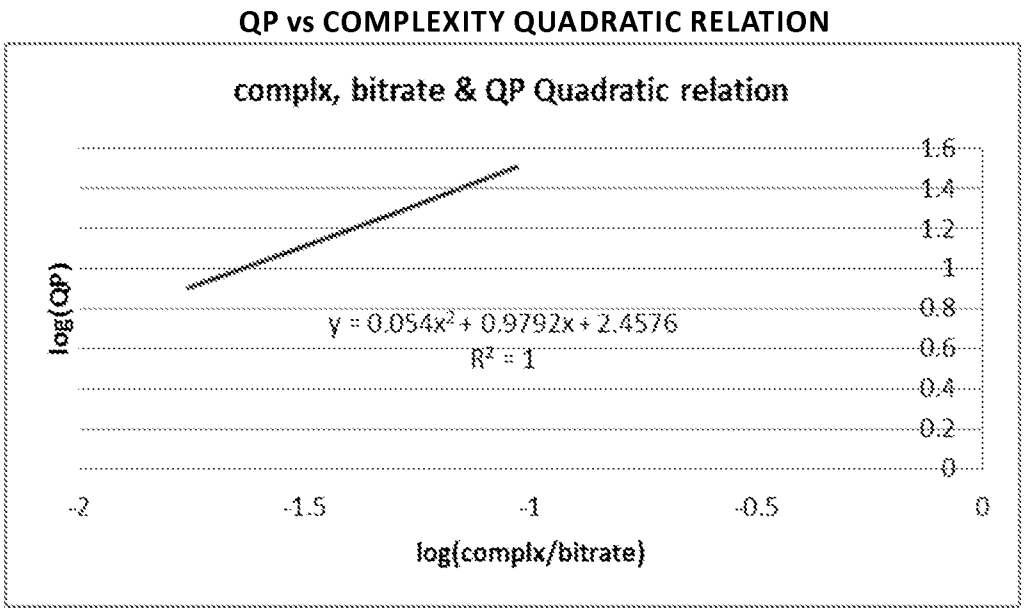


FIG. 14

bpp to b to bpp_id mapping

0	1	2	3	4	5	6	7	8
0.01052	0.02104	0.04208	0.08416	0.16832	0.33664	0.67328	1.34656	2.69312

FIG. 15

cpx to cpx_id mapping

0	1	2	3	4	5	6	7	8	9
0.75	1.5	2.25	3	4	5	6	7.5	9.25	INT_MAX

FIG. 16

MEAN QP TABLE

	0	1	2	3	4	5	6	7	8
0	28	22	16	12	8	6	4	2	1
1	30	24	20	16	12	8	6	4	2
2	34	28	22	18	14	10	7	5	3
3	38	32	26	20	16	12	8	6	4
4	44	38	32	26	20	14	10	7	5
5	48	42	36	30	24	18	12	8	6
6	50	44	38	32	26	20	14	9	7
7	52	46	40	34	28	22	16	10	8
8	54	48	42	36	30	24	18	12	9
9	56	50	44	38	32	26	20	14	10

FIG. 17

MINIMUM QP TABLE

	0	1	2	3	4	5	6	7	8
0	24	18	12	9	6	4	2	1	1
1	26	20	16	12	9	5	3	2	1
2	30	22	18	14	11	7	5	3	2
3	34	28	22	16	12	9	5	3	2
4	40	34	28	22	16	11	7	5	3
5	44	38	32	26	20	14	9	5	3
6	46	40	34	28	22	16	11	6	4
7	48	42	36	30	24	18	12	7	5
8	50	44	38	32	26	20	14	9	6
9	52	46	40	34	28	22	16	11	7

FIG. 18

MAXIMUM QP TABLE

	0	1	2	3	4	5	6	7	8
0	32	26	20	16	11	9	6	4	3
1	34	28	24	20	16	11	9	6	4
2	38	32	26	22	18	14	10	8	6
3	42	36	30	24	20	16	11	9	7
4	48	42	36	30	24	18	14	10	8
5	52	46	40	34	28	22	16	14	9
6	54	48	42	36	30	24	18	14	10
7	56	50	44	38	32	26	20	16	11
8	58	52	46	40	34	28	22	16	12
9	60	54	48	42	36	30	24	18	14

FIG. 19

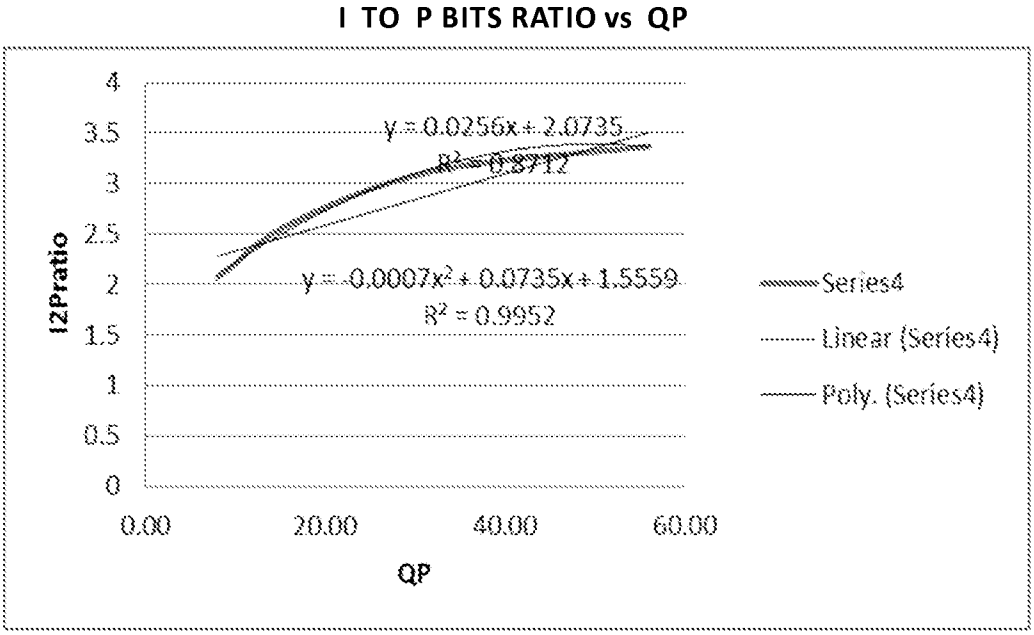


FIG. 20

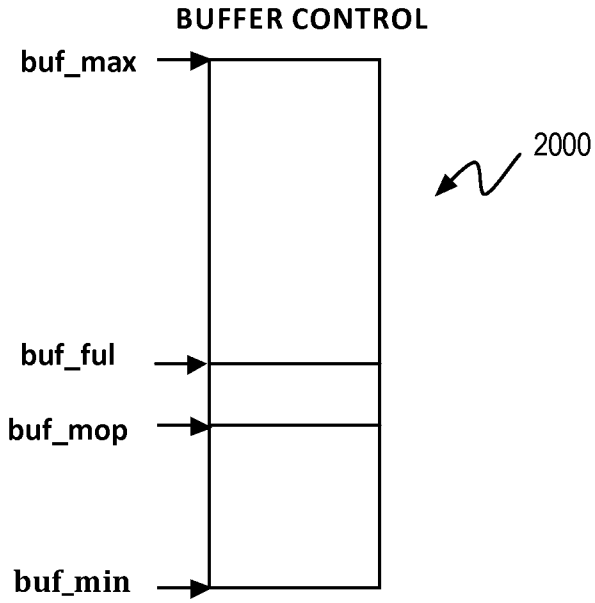


FIG. 21

THRESHOLD FOR RATE OF BUF INCREMENT

0	1	2	3	4	5	6	7	8	9	10
0.1	0.09	0.08	0.07	0.06	0.05	0.04	0.03	0.02	0.02	0.02

FIG. 22A

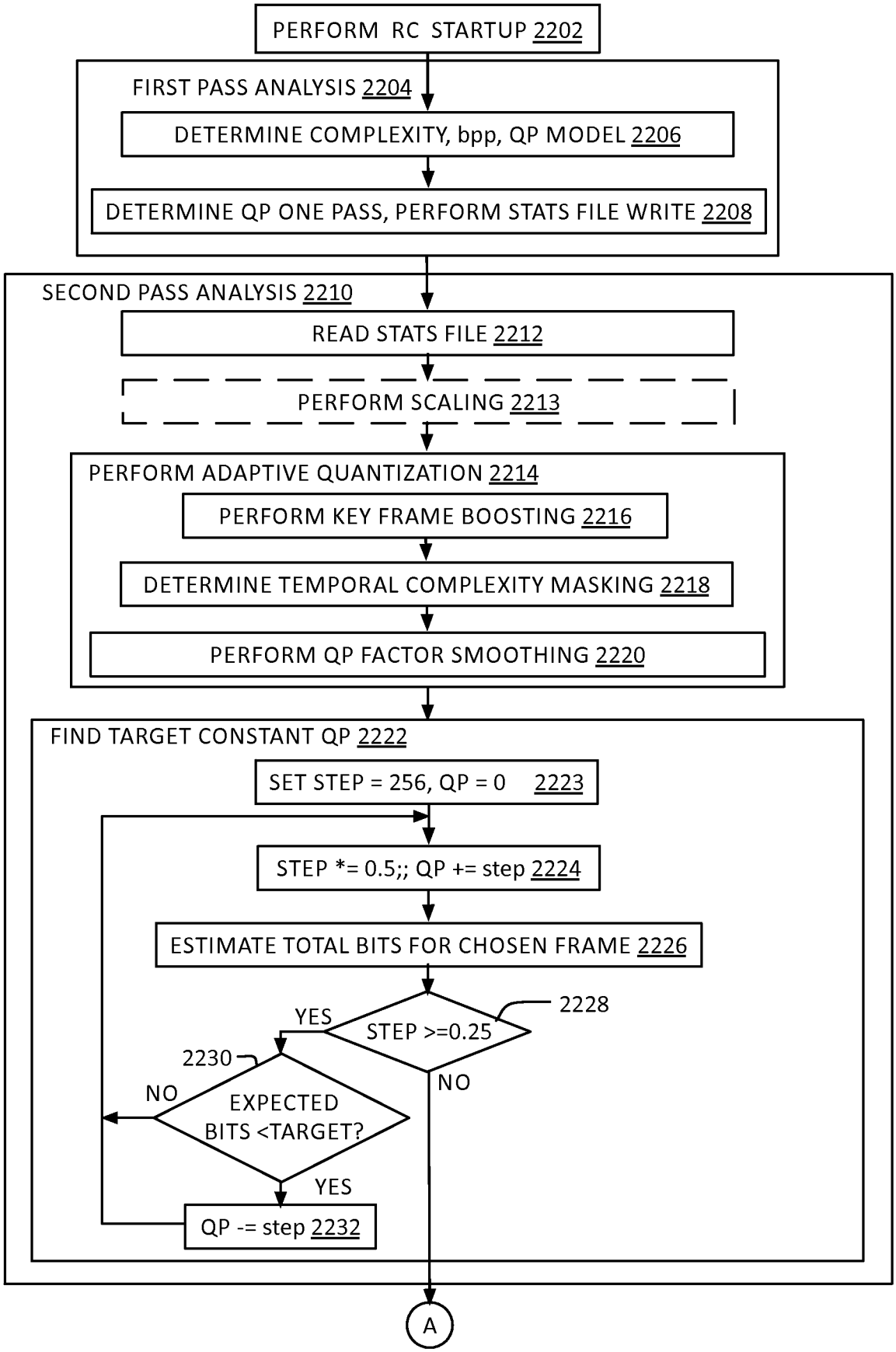
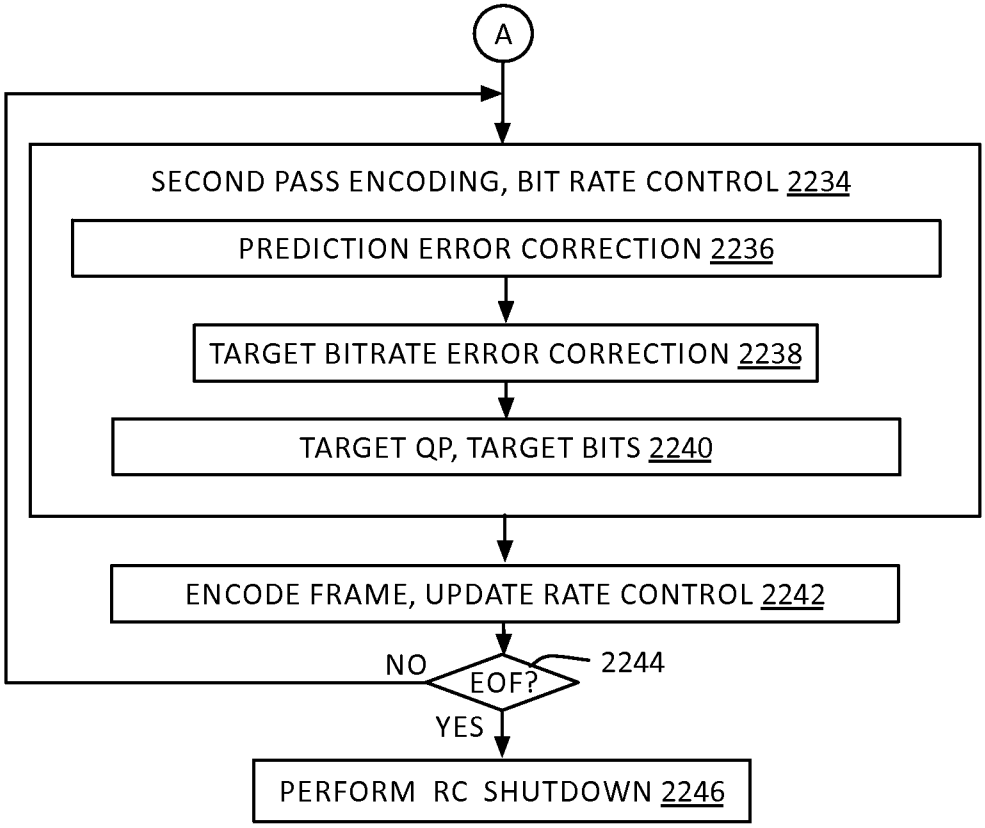

PERFORM RC STARTUP 2202
FIRST PASS ANALYSIS 2204
DETERMINE COMPLEXITY, bpp, QP MODEL 2206
DETERMINE QP ONE PASS, PERFORM STATS FILE WRITE 2208
SECOND PASS ANALYSIS 2210
READ STATS FILE 2212
PERFORM SCALING 2213
PERFORM ADAPTIVE QUANTIZATION 2214
PERFORM KEY FRAME BOOSTING 2216
DETERMINE TEMPORAL COMPLEXITY MASKING 2218
PERFORM QP FACTOR SMOOTHING 2220
FIND TARGET CONSTANT QP 2222
SET STEP = 256, QP = 0 2223
STEP *= 0.5;; QP += step 2224
ESTIMATE TOTAL BITS FOR CHOSEN FRAME 2226
2228
YES
STEP >=0.25
NO
2230
NO
EXPECTED BITS <TARGET?
YES
QP -= step 2232
A

FIG. 22B


A
SECOND PASS ENCODING, BIT RATE CONTROL 2234
PREDICTION ERROR CORRECTION 2236
TARGET BITRATE ERROR CORRECTION 2238
TARGET QP, TARGET BITS 2240
ENCODE FRAME, UPDATE RATE CONTROL 2242
NO
EOF?
2244
YES
PERFORM RC SHUTDOWN 2246

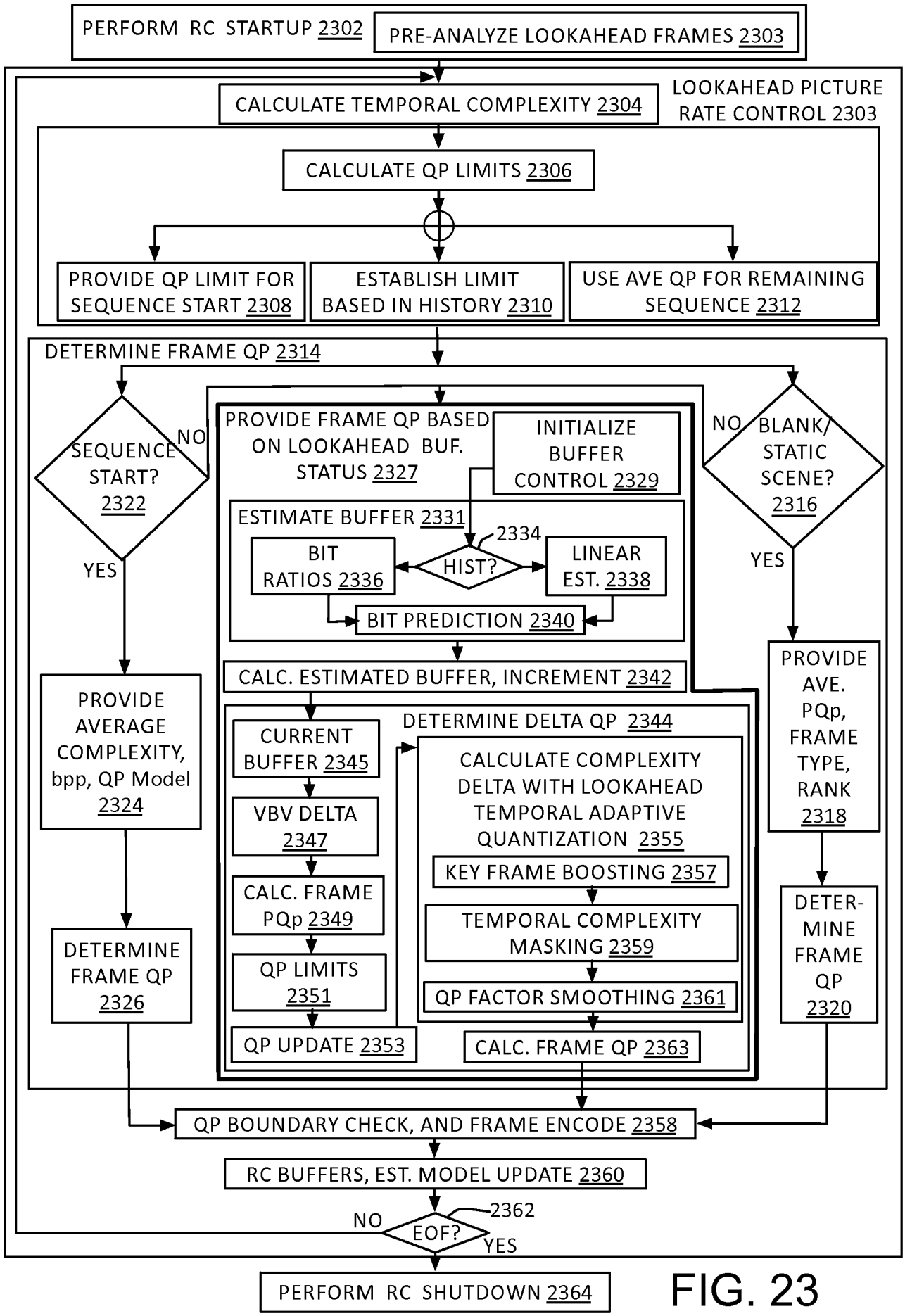

PERFORM RC STARTUP 2302
PRE-ANALYZE LOOKAHEAD FRAMES 2303
LOOKAHEAD PICTURE RATE CONTROL 2303
CALCULATE TEMPORAL COMPLEXITY 2304
CALCULATE QP LIMITS 2306
PROVIDE QP LIMIT FOR SEQUENCE START 2308
ESTABLISH LIMIT BASED IN HISTORY 2310
USE AVE QP FOR REMAINING SEQUENCE 2312
DETERMINE FRAME QP 2314
SEQUENCE START? 2322
NO
YES
PROVIDE FRAME QP BASED ON LOOKAHEAD BUF. STATUS 2327
INITIALIZE BUFFER CONTROL 2329
ESTIMATE BUFFER 2331
2334
BIT RATIOS 2336
HIST?
LINEAR EST. 2338
BIT PREDICTION 2340
CALC. ESTIMATED BUFFER, INCREMENT 2342
DETERMINE DELTA QP 2344
CURRENT BUFFER 2345
VBV DELTA 2347
CALC. FRAME PQp 2349
QP LIMITS 2351
QP UPDATE 2353
CALCULATE COMPLEXITY DELTA WITH LOOKAHEAD TEMPORAL ADAPTIVE QUANTIZATION 2355
KEY FRAME BOOSTING 2357
TEMPORAL COMPLEXITY MASKING 2359
QP FACTOR SMOOTHING 2361
CALC. FRAME QP 2363
BLANK/ STATIC SCENE? 2316
NO
YES
PROVIDE AVERAGE COMPLEXITY, bpp, QP Model 2324
DETERMINE FRAME QP 2326
PROVIDE AVE. PQp, FRAME TYPE, RANK 2318
DETER-MINE FRAME QP 2320
QP BOUNDARY CHECK, AND FRAME ENCODE 2358
RC BUFFERS, EST. MODEL UPDATE 2360
2362
NO
EOF?
YES
PERFORM RC SHUTDOWN 2364

FIG. 23

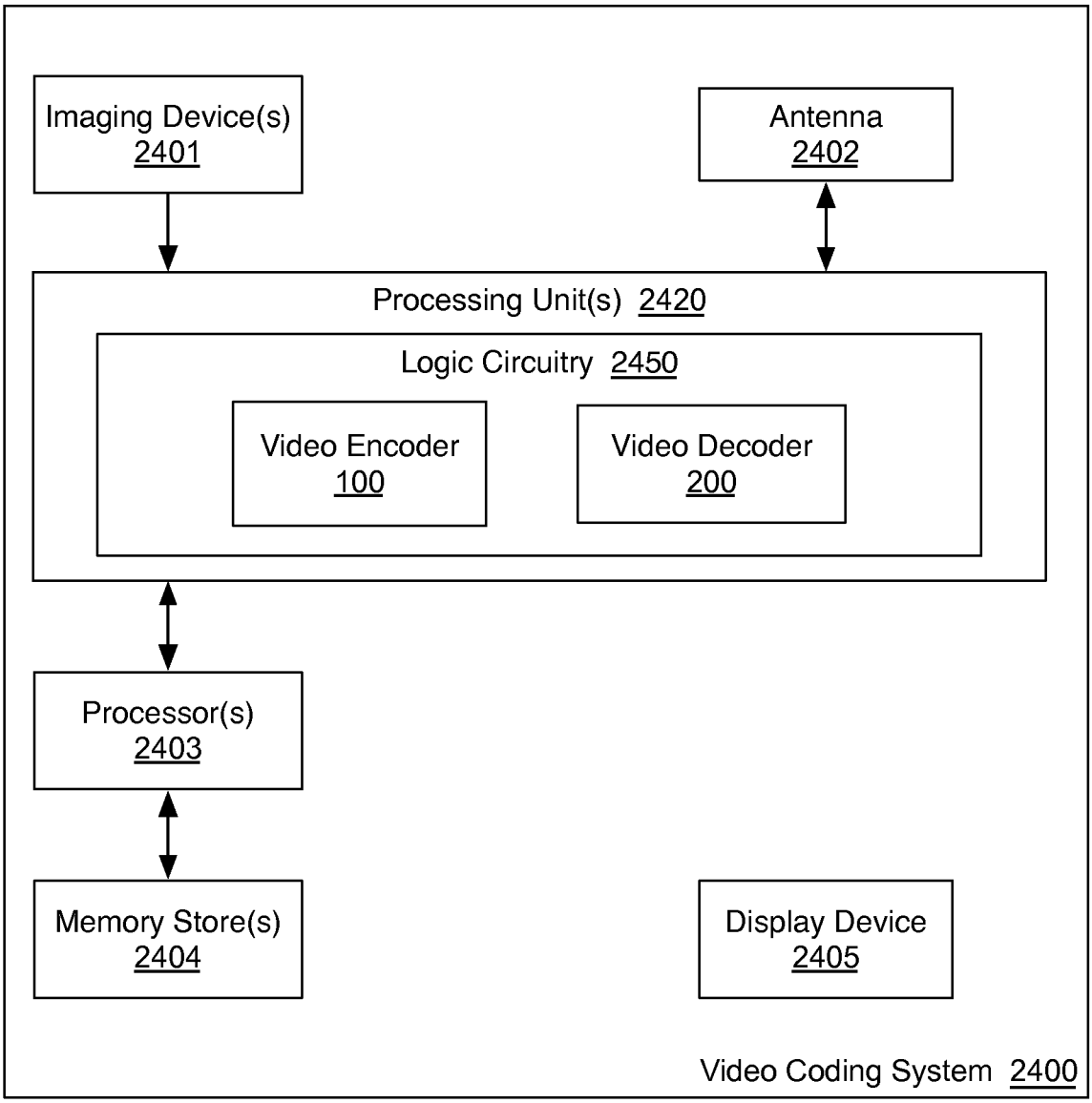

Imaging Device(s)
2401
Antenna
2402
Processing Unit(s) 2420
Logic Circuitry 2450
Video Encoder
100
Video Decoder
200
Processor(s)
2403
Memory Store(s)
2404
Display Device
2405
Video Coding System 2400

FIG. 24

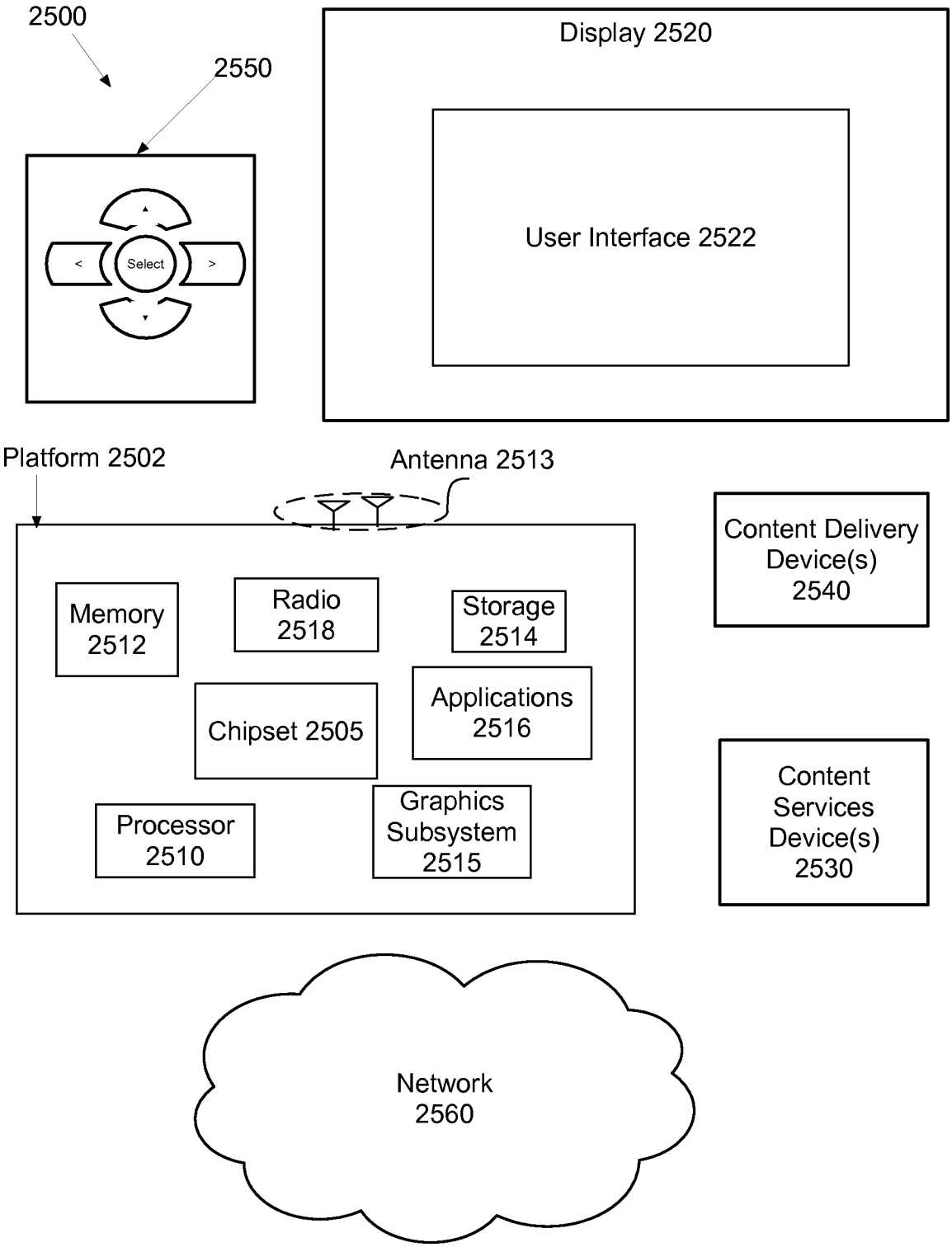

2500
2550
Select
Display 2520
User Interface 2522
Platform 2502
Antenna 2513
Memory 2512
Radio 2518
Storage 2514
Chipset 2505
Applications 2516
Processor 2510
Graphics Subsystem 2515
Content Delivery Device(s) 2540
Content Services Device(s) 2530
Network 2560

FIG. 25

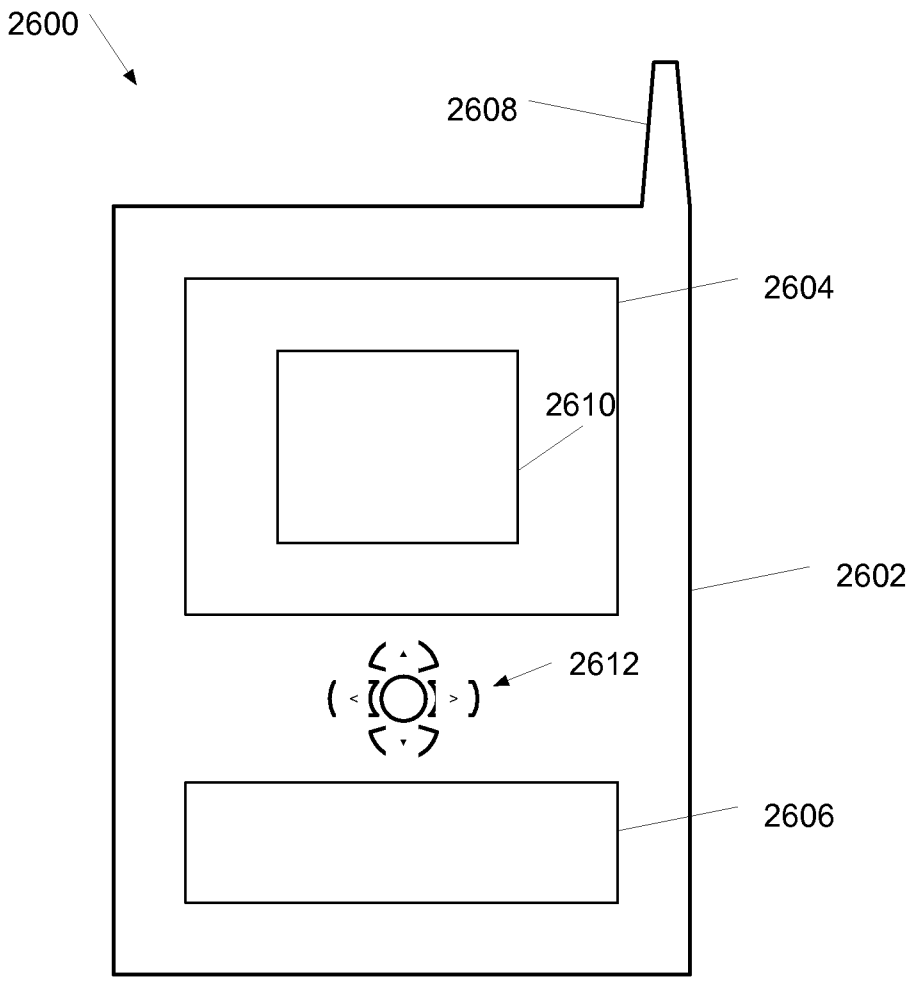


FIG. 26

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/013915

A. CLASSIFICATION OF SUBJECT MATTER

H04N 19/136(2014.01)i, H04N 21/2662(2011.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04N 19/136; none ; H04N 7/26; H04N 7/32; H04B 1/66; H04N 21/2662

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: hierarchy, bitrate, context, quantization, level, picture, rank, reference and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2008-0232459 A1 (CHEUNG AUYEUNG) 25 September 2008 See paragraphs [0069]-[0070], [0083]; figures 1, 3; and claims 1, 6.	1-55
A	US 2009-0103610 A1 (ATUL PURI) 23 April 2009 See abstract; paragraphs [0066], [0090]-[0092], [0120]; figure 6A; and claim 1.	1-55
A	US 2012-0082214 A1 (MICHAEL HOROWITZ et al.) 5 April 2012 See paragraphs [0037]-[0043]; figure 2; and claims 17-18.	1-55
A	KR 10-2008-0107867 A (SAMSUNG ELECTRONICS CO., LTD.) 11 December 2008 See paragraphs [0014]-[0017]; figure 5; and claim 1.	1-55
A	WO 2009-158113 A2 (MICROSOFT CORPORATION) 30 December 2009 See paragraphs [0076]-[0079]; figure 4; and claim 1.	1-55

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
20 May 2014 (20.05.2014)	**21 May 2014 (21.05.2014)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea
Facsimile No. +82-42-472-7140

Authorized officer
KIM, Seong Woo
Telephone No. +82-42-481-3348

Form PCT/ISA/210 (second sheet) (July 2009)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2014/013915

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008-0232459 A1	25/09/2008	US 8396118 B2	12/03/2013
US 2009-0103610 A1	23/04/2009	US 2005-180502 A1	18/08/2005
		US 7492820 B2	17/02/2009
		US 8036267 B2	11/10/2011
US 2012-0082214 A1	05/04/2012	WO 2012-047304 A1	12/04/2012
KR 10-2008-0107867 A	11/12/2008	None	
WO 2009-158113 A2	30/12/2009	CN 102057677 A	11/05/2011
		CN 102057677 B	02/10/2013
		CN 103428497 A	04/12/2013
		EP 2283655 A2	16/02/2011
		EP 2283655 A4	27/07/2011
		JP 2011-524130 A	25/08/2011
		JP 2013-225889 A	31/10/2013
		KR 10-2011-0015002 A	14/02/2011
		MX 2010012818 A	21/12/2010
		US 2009-0296808 A1	03/12/2009
		WO 2009-158113 A3	04/03/2010