

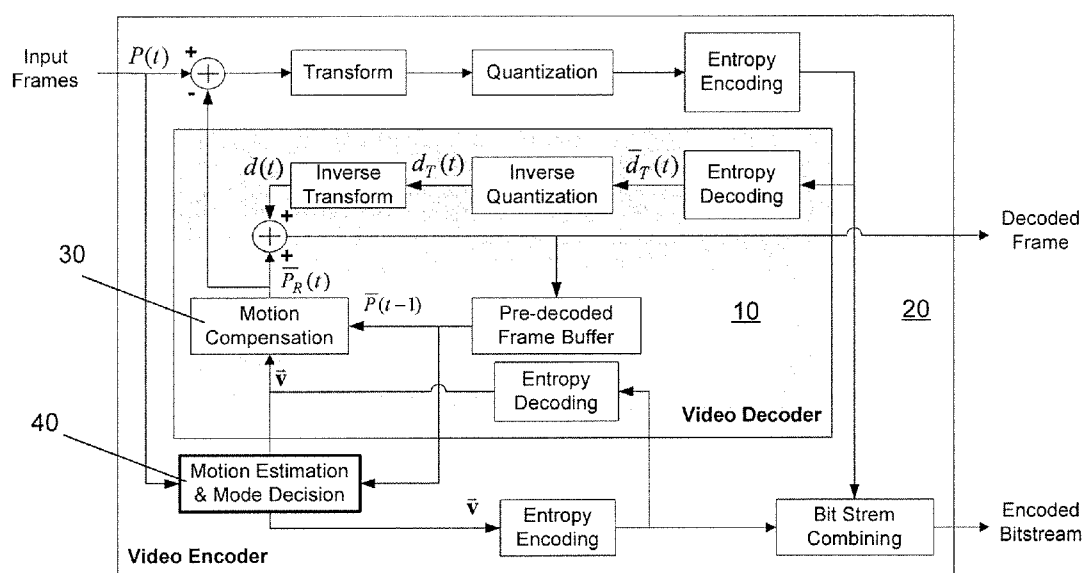


FIG. 1

210

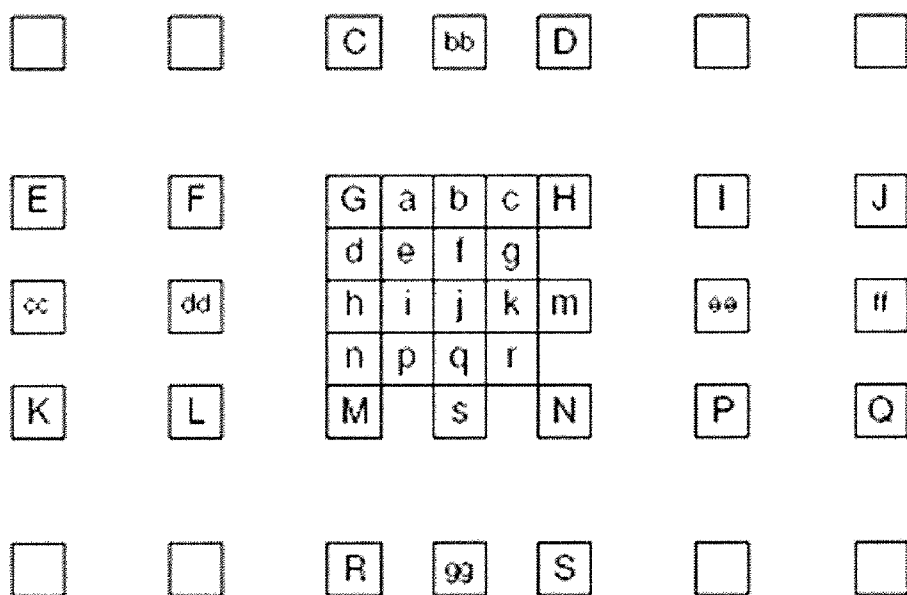


FIG. 2

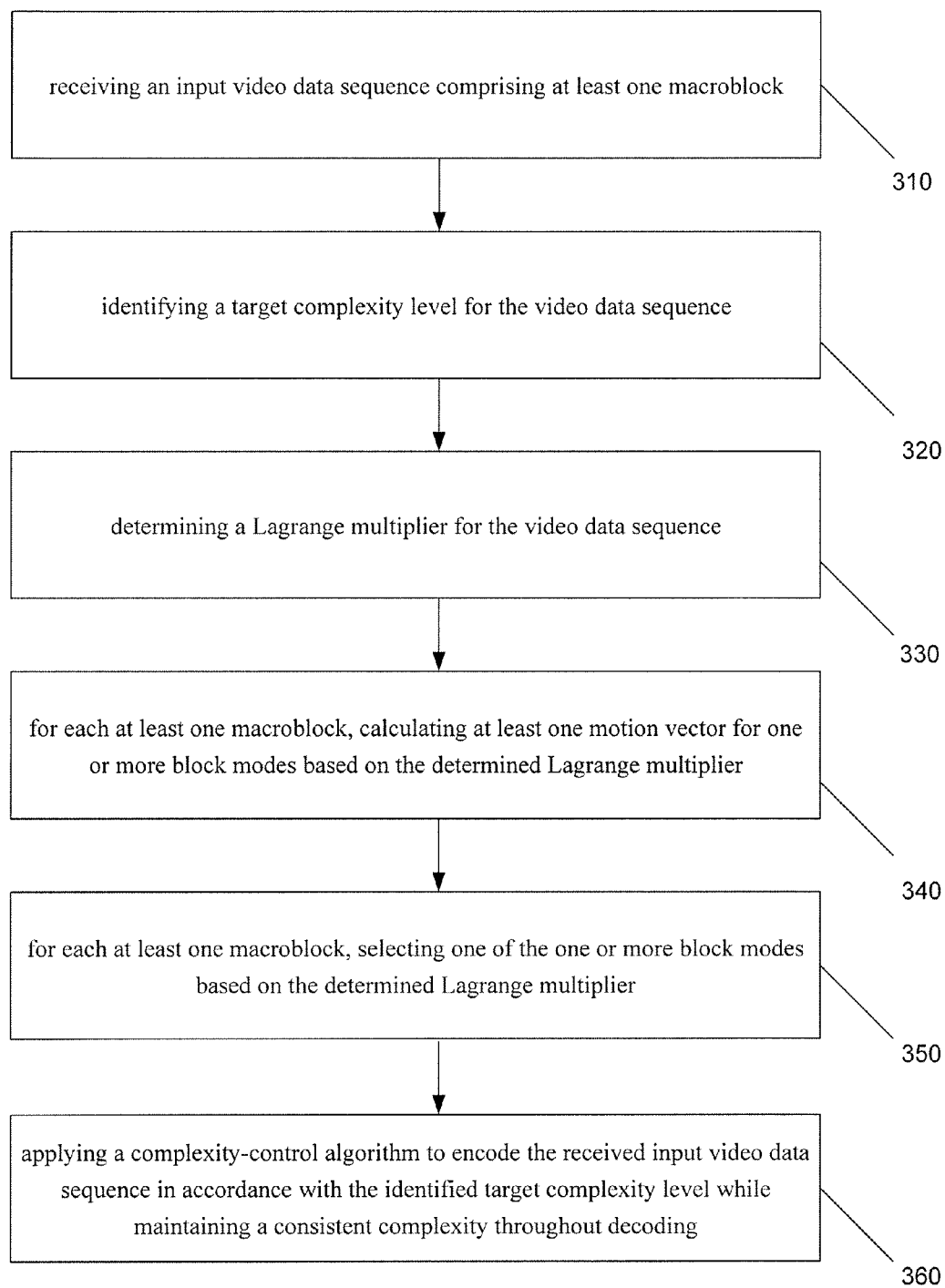


FIG. 3

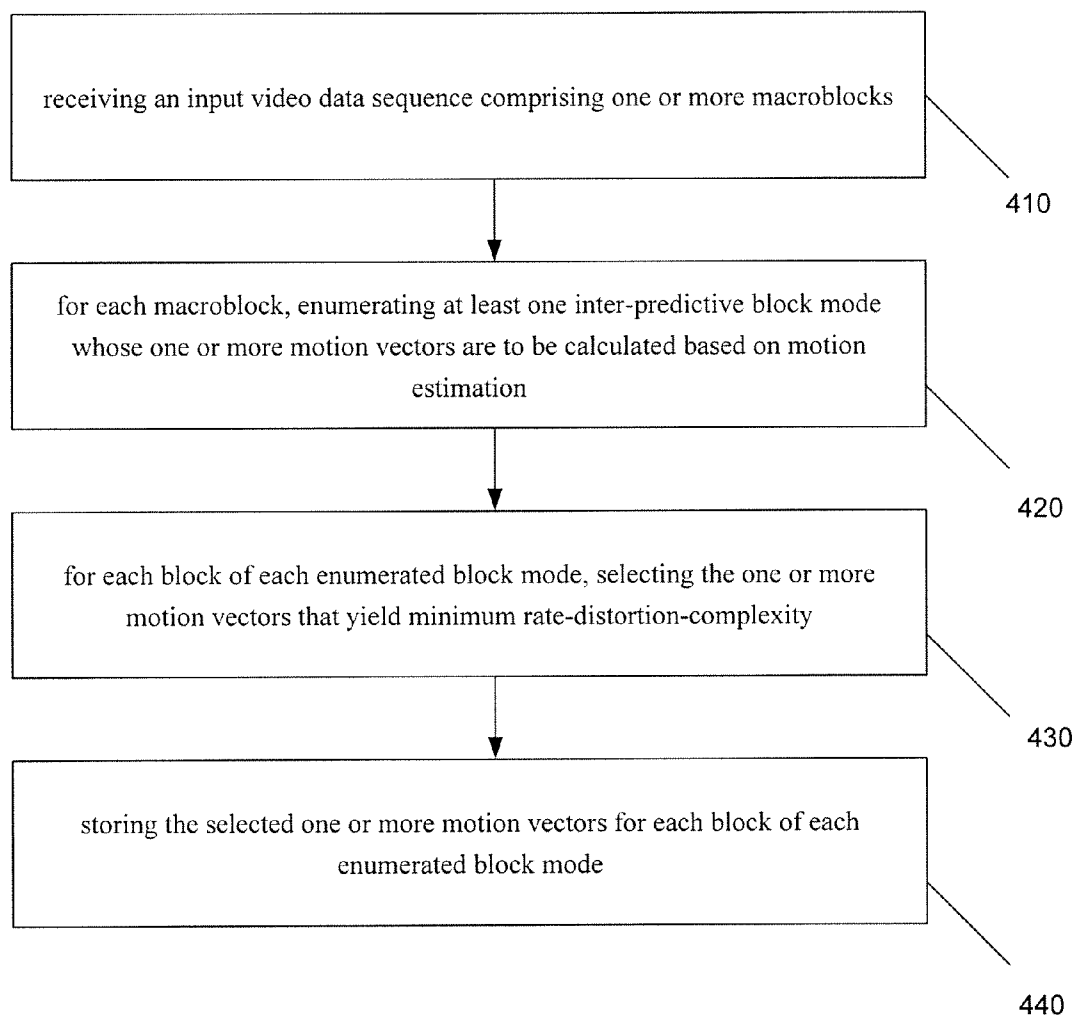


FIG. 4

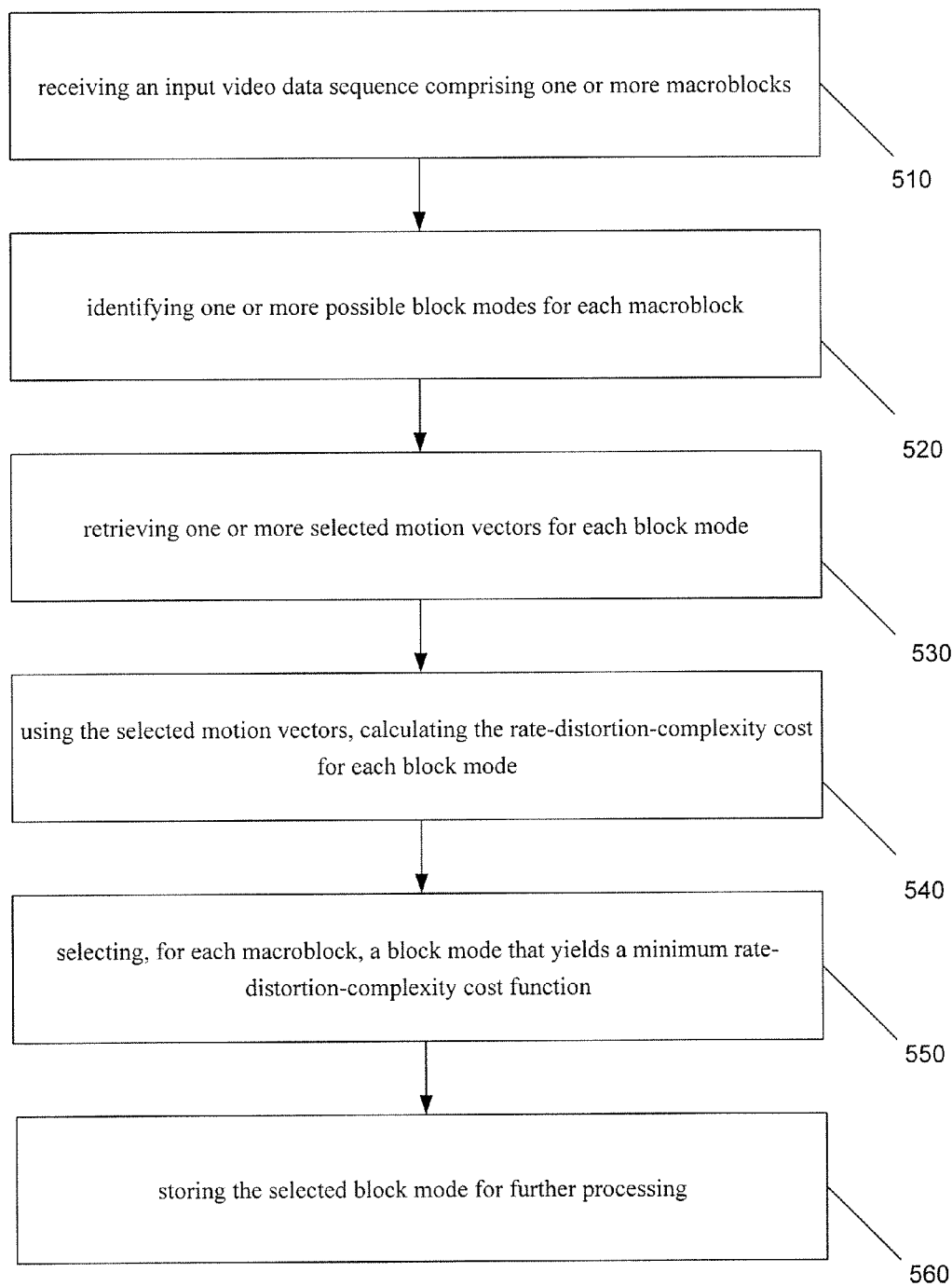


FIG. 5

SYSTEM AND METHOD FOR MOTION ESTIMATION AND MODE DECISION FOR LOW-COMPLEXITY H.264 DECODER

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is based on U.S. provisional patent application Ser. No. 60/658,440, filed Mar. 4, 2005, which is incorporated herein by reference for all purposes and from which priority is claimed.

BACKGROUND OF THE INVENTION

[0002] 1. Technical Field

[0003] The present invention relates to systems and methods for motion estimation and mode decision for low-complexity H.264 standard encoders/decoders.

[0004] 2. Background Art

[0005] Emerging video coding standards like H.264 achieve significant advances in improving video quality and reducing bandwidth, but generally at the cost of greatly increased computational complexity at both the encoder and the decoder. Playing encoded videos produced by such compression standards requires substantial computational resources and thus results in substantial power consumption. This may be a serious concern in power-sensitive applications, such as handheld devices and other devices used in mobile applications.

[0006] Many portable media application devices such as mobile handheld devices are becoming increasingly popular. The computational resources available on these handheld devices is becoming relatively scarce as applications of increasing complexity and number are operated by the devices. Accordingly, there is growing interest in complexity-aware/power-aware video coding solutions.

[0007] Most of today's video coding systems encode video bit streams to achieve the best video quality (e.g., the minimal signal distortion) while satisfying certain bitrate constraints. Specifically the following optimization problem formulation is often adopted:

$$\min_P D(P) \text{ s.t., } R(P) \leq R_T \quad (1)$$

where P represents the control variables (CV) which eventually determine the final video quality and bit rate. Typical CVs include quantization parameter (QP), motion vector, motion estimation block mode, etc. D is the distortion introduced by the encoding process. R is the bit rate of the encoded video and R_T is the target bit rate. The solution of the above problem aims at identifying the optimal control variables for each coding unit in order to minimize the average distortion while satisfying the bit rate constraint. Though in practice, some design choices for the control variables may be made based on real-world resource limitations (e.g., memory and computational complexity), Equation (1) does not explicitly model this required complexity in video encoding or decoding. As a matter of fact, many recent advances in coding efficiency are accomplished by using increasingly complex computational modules.

[0008] Methods for reducing computational complexity in the prior art include ARMS and National Semiconductor develop a systematic approach called PowerWise technology,

which can efficiently reduce the power consumption of mobile multimedia applications through adaptive voltage scaling (AVS). (See National's PowerWise™ technology, described at <http://www.national.com/appinfo/power/powerwise.html>, which is fully incorporated herein by reference). Zhou et al. implements an H.264 decoder based on Intel's single-instruction-multiple-data (SIMD) architecture that reduces the decoding complexity and improved the H.264 decoding speed by up to three times. (See X. Zhou, E. Li, and Y.-K. Chen, "Implementation of H.264 Decoder on General-Purpose Processors with Media Instructions", in Proc. of SPIE Visual Communications and Image Processing, Jan. 2003, which is fully incorporated herein by reference). Ray and Radha propose a method to reduce the decoding complexity by selectively replacing the I-B-P Group of Pictures (GOP) structure with one using I-P only. (See A. Ray and H. Radha, "Complexity-Distortion Analysis of H.264/JVT Decoder on Mobile Devices," Picture Coding Symposium (PCS), December 2004, which is fully incorporated herein by reference). Lengwehasatit and Ortega developed a method to reduce the decoding complexity by optimizing the Inverse DCT implementation. (See K. Lengwehasatit and A. Ortega, "Rate Complexity Distortion Optimization for Quadtree-Based DCT Coding", ICIP 2000, Vancouver, BC, Canada, September 2000, which is fully incorporated herein by reference). He et al. optimizes the power-rate-distortion performance by constraining the sum of absolute difference (SAD) operations during the motion estimation process at the encoder. (See Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-Rate-Distortion Analysis for Wireless Video Communication under Energy Constraints," IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Integrated Multimedia Platforms, 2004, which is fully incorporated herein by reference). In addition, power-aware joint source channel coding is also an active topic for mobile wireless video communication. (See Y. Eisenberg, C. E. Luna, T. N. Pappas, R. Berry, A. K. Katsaggelos, Joint source coding and transmission power management for energy efficient wireless video communications, *CirSysVideo*(12), No. 6, June 2002, pp. 411-424; Q. Zhang, W. Zhu, Zu Ji, and Y. Zhang, "A Power-Optimized Joint Source Channel Coding for Scalable Video Streaming over Wireless Channel", IEEE International Symposium on Circuits and Systems (ISCAS) 2001, May, 2001, Sydney, Australia; X. Lu, E. Erkip, Y. Wang and D. Goodman, "Power efficient multimedia communication over wireless channels", IEEE Journal on Selected Areas on Communications, Special Issue on Recent Advances in Wireless Multimedia, Vol. 21, No. 10, pp. 1738-1751, Dec., 2003, all of which are fully incorporated herein by reference). Unlike the conventional paradigm using complex encoding and light decoding, Girod et al. propose a distributed video coding system which transfers the motion estimation process from the encoder to the decoder so that the encoding complexity can be greatly reduced. (See B. Girod, A. Aaron, S. Rane and D. Rebollo-Monedero, "Distributed video coding," *Proc. of the IEEE*, Special Issue on Video Coding and Delivery, 2005, which is fully incorporated herein by reference).

[0009] Furthermore, the computational complexity of each component of a video decoding system varies. Some are relatively constant and independent of the encoded data while others heavily depend on the coding results. For example, the components of inverse quantization and inverse transform have nearly fixed computational cost per coding unit while the motion compensation component has variable complexity

depending on the block mode and the type of motion vector. Furthermore, the decoder complexity is dominated by the interpolation filtering process used in motion compensation if the motion vectors are sub-pixel. Other parts of the decoding system, like entropy decoding and inverse transform, do not incur significant computational cost when compared to the interpolation process.

[0010] As noted, motion estimation is usually the most computationally complex process since it involves searching over a large range of possible reference locations, each of which may require interpolation filtering. Among the components in the decoding system, the interpolation procedure used in the motion compensation component consumes the most computational resources (about 50%) due to the use of sub-pixel motion vectors. Accordingly, one way to increase power consumption efficiency in video decoding would be to reduce the major computational cost of the motion compensation interpolation procedure.

[0011] Many fast motion estimation algorithms have been developed to reduce the motion estimation complexity during encoding. (See A. M. Tourapis, "Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation," Proceedings of Visual Communications and Image Processing 2002 (VCIP-2002), San Jose, Calif., Jan 2002, pp. 1069-79; H.-Y. Cheong, A. M. Tourapis, "Fast Motion Estimation within the H.264 codec," in proceedings of ICME-2003, Baltimore, MD, Jul. 6-9, 2003, both of which are incorporated herein by reference). Other work proposes scalable methods for motion estimation to control the coding complexity. (See M. Schaar, H. Radha, Adaptive motion-compensation fine-granular-scalability (AMC-FGS) for wireless video, IEEE Trans. on CSVT, vol. 12, no. 6, 360-371, 2002, which is incorporated herein by reference). Nevertheless these methods all focus on the encoding complexity reduction instead of the decoding complexity.

[0012] Accordingly, there exists a need in the art for an improved system and method for video encoding/decoding with improved motion estimation which reduces computational costs and power consumption in the decoder.

SUMMARY OF THE INVENTION

[0013] The objects of the present invention may be met with a novel system and method for optimizing the selection of the motion vectors and motion compensation block modes in video decoding to reduce the computational cost of decoding while maintaining the desired video quality.

[0014] The present invention meets needs in the prior art by providing a system and method for optimizing a video encoder, including the steps of receiving an input video data sequence which includes at least one macroblock, identifying a target complexity level for the video data sequence, determining a Lagrange multiplier for the video data sequence, for each macroblock, calculating at least one motion vector for each block mode based on the determined Lagrange multiplier, for each at least one macroblock, selecting one of the block modes based on the determined Lagrange multiplier, and applying a complexity-control algorithm to encode the received input video data sequence in accordance with the identified target complexity level while maintaining a consistent complexity throughout decoding of the input video data sequence.

[0015] In another exemplary embodiment, the present invention provides a system and method for optimizing a video encoder, including the steps of receiving an input video

data sequence including one or more macroblocks, for each macroblock, enumerating at least one inter-predictive block mode whose one or more motion vectors are to be calculated based on motion estimation, for each block of each enumerated block mode, selecting the one or more motion vectors that yield minimum rate-distortion-complexity, and storing the selected one or more motion vectors for each block of each enumerated block mode.

[0016] In another exemplary embodiment, the present invention provides a system and method for optimizing a video encoder, including the steps of receiving an input video data sequence including one or more macroblocks, identifying one or more possible block modes for each macroblock, retrieving one or more selected motion vectors for each block mode, using the selected motion vectors, calculating the rate-distortion-complexity cost for each block mode, selecting, for each macroblock, a block mode that yields a minimum rate-distortion-complexity cost function, and storing at least one of the selected block modes for further processing.

[0017] The accompanying drawings, which are incorporated and constitute part of this disclosure, illustrate preferred embodiments of the invention and serve to explain the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

[0018] FIG. 1 is a conceptual diagram of an exemplary video coding system;

[0019] FIG. 2 is an exemplary diagram showing sub-pixel locations within a macroblock for exemplary purposes;

[0020] FIG. 3 is a flow chart describing the steps of a method in accordance with an exemplary embodiment of the present invention;

[0021] FIG. 4 is a flow chart describing the steps of a method in accordance with an exemplary embodiment of the present invention;

[0022] FIG. 5 is a flow chart describing the steps of a method in accordance with an exemplary embodiment of the present invention.

[0023] Throughout the Figures, the same reference numerals and characters, unless otherwise stated, are used to denote like features, elements, components or portions of the illustrated embodiments. Moreover, while the present invention will now be described in detail with reference to the Figures, it is done so in connection with the illustrative embodiments.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0024] The present invention relates to an important aspect of the complexity minimization problem in video decoding—i.e., developing an encoding algorithm that achieves both high video quality and low decoding complexity while satisfying the bit rate constraint. The object is to reduce the complexity requirement of emerging video codecs, such as H.264, particularly on resource-limited devices such as handheld devices. The present invention is different from the approaches described above in that the present invention modifies the video encoding algorithm to minimize the required complexity at the decoder, not the encoder. The approach does not require substantial modification to existing decoder implementations. The present invention may be implemented such that it modifies the non-normative parts of, e.g., the H.264 encoding algorithm, such that it is compatible with standards-compliant decoders. However, the present

invention is not limited to H.264, but may be implemented in accordance with any video encoding/decoding system, including systems operating in accordance with the MPEG-4 standard, the Motion Compensated Embedded Zero Block Coding ("MC-EZBC") standard, and others.

[0025] Additionally, other techniques for the decoder power minimization, such as those discussed above, are complementary and can be used in conjunction with the present invention. Notably, the exemplary system as described herein below may be implemented in accordance with the H.264 protocol. However, the present invention is not limited to H.264, but may be implemented in accordance with any video encoding/decoding system, including systems operating in accordance with the MPEG-4 standard and the Motion Compensated Embedded Zero Block Coding ("MC-EZBC") standard.

[0026] By way of background, generally, when considering the decoder's complexity during video encoding, the optimization problem is formulated as follows.

$$\min_P D(P) \text{ s.t., } R(P) \leq R_T, C(P) \leq C_T \quad (2)$$

[0027] where C is the computational complexity at the decoder. Compared with the problem defined in Equation (1), a constraint on computational complexity is explicitly added. The solution for Equation (2) generally determines the best control variables, P , for each coding unit. Similar to the case for Equation (1), the control variables include quantization parameter, block mode of the motion compensation process, and the associated motion vectors.

[0028] Among the control variables, the motion vectors have the largest impact on the decoding complexity. Motion vectors can be of integer or fractional values corresponding to a displacement distance of integral pixels or fractional pixels. When a motion vector is of a sub-pixel value, multi-tap filtering is required to compute interpolation to form a reference block that is needed in the motion compensation process in the decoder. Such interpolation filtering involves huge computational cost and typically significantly increases the overall decoding complexity (the interpolation component constitutes about 50% of the decoding complexity). Although for mobile multimedia applications there are other power consuming components like wireless communication, display, and memory access, the decoding process is typically a significant one. Therefore improving the cost associated with the interpolation process is important for implementing a low-power decoding system, either in hardware or software.

[0029] In accordance with an exemplary embodiment of the present invention, the conventional rate-distortion framework is extended based on the Lagrange optimization method to incorporate the computational complexity. To estimate the complexity associated with different types of motion vectors, models are used to approximate the implementation cost involved in the interpolation filtering process.

[0030] In addition, in accordance with an exemplary embodiment of the present invention, the rate control algorithm may be extended to handle the joint rate-complexity control issue so that both the targets of rate and complexity can be met. This optimization method intelligently selects the block mode and motion vector type of each coding unit to achieve the highest video quality.

[0031] Referring now to FIG. 1, an exemplary diagram for a typical hybrid motion compensation and block-transform video coding system is shown. The interior box components 10 include the decoding portion (which typically is also simulated in the encoder system for rate control purpose). Additional components of a companion encoder system are shown in exterior box components 20. The present invention focuses generally on improvements in the motion compensation functions 30 by way of the motion estimation and mode decision functions 40. A general discussion of video encoding/decoding principles follows herein, followed by a discussion of these principles as modified in accordance with exemplary embodiments of the present invention.

[0032] Further referring to FIG. 1, the basic decoding unit in a video decoder is a macroblock (MB), which is an array of spatially-adjacent pixels in a video sequence. For exemplary purposes, FIG. 2 is a diagram showing sub-pixel locations within a macroblock.

[0033] For each MB, the encoded bit stream may first undergo entropy decoding to obtain the syntax bits (this step is not shown in the figure, but would be known to one of ordinary skill in the art), motion vector V , and quantized coefficients $\bar{d}_T(t)$, where t is the time index of the image frame. Typical entropy codecs include variable length coding (VLC) and adaptive arithmetical coding (AAC). Inverse quantization is then employed to obtain the transform coefficient $d_T(t)$, which is further fed to an inverse transform module to reconstruct the pixel value or prediction error $d(t)$, depending on whether intro- or inter-coded mode is utilized during encoding. For inter-coding mode, motion compensation is applied to generate the reference image $\bar{P}_R(t)$ using motion vector v and previously decoded and buffered reference image $\bar{P}(t-1)$. Motion compensation refers to the process of compensating the image displacement due to motion across frames. When the motion vector is of a sub-pixel value, interpolation is needed to compute the reference image. Lastly, by combining the prediction error $d(t)$ and the reference image $\bar{P}_R(t)$ the decoded image of the current frame is output.

[0034] The basic concept of motion estimation is to search for an optimal block with similar values in previous coded frames as the reference signal for the block in current frame so that the encoding cost can be minimized. The optimal reference signal position is indicated by the displacement vector, called a motion vector. Motion estimation applies the basic idea of inter-frame predictive coding. Sometimes, multiple reference signals are used to form motion estimation, for example in bidirectional inter-frame prediction. Motion vectors are entropy encoded in a differential and predictive manner. (See T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, "Overview of the H.264/AVC Video Coding Standard," IEEE Trans. Circuits Syst. Video Technol., vol 13, pp. 560-576. Jul. 2003, which is incorporated herein by reference).

[0035] Compared to motion estimation, motion compensation is the procedure by which the decoder extracts a reference signal from the location indicated by the motion vector. In reconstructing the reference signal, interpolation is a widely adopted technique used to improve the compensation precision when the motion vector has a sub-pixel value. The effectiveness of the sub-pixel motion compensation has been verified in H.263 and subsequent coding standards, at the cost

of increasing complexity. Therefore reducing the motion compensation complexity is the most important target for improvement.

[0036] H.264 uses up to quarter pixel precision during interpolation. (See T. Wedi; H. G. Musmann, Motion- and aliasing-compensated prediction for hybrid video coding-Page(s): IEEE Trans. Circuits Syst. Video Technol., vol 13, pp. 577-586. Jul. 2003, which is incorporated herein by reference). All half-pixel locations undergo 6-tap FIR filtering horizontally and vertically, whenever any one applies. All quarter-pixel locations undergo 2-tap average filtering using integer and half pixels. For example, the following formulae are used to calculate sub pixel b and e:

$$b = ((E - 5F + 20G + 20H - 5I - J) + 16) / 32$$

$$e = (b + h + 1) / 2$$

[0037] The amount of filtering varies depending on the exact location of the pixel. Referring again to FIG. 2, Table 1 below lists the possible interpolation operations and their associated complexities:

TABLE 1

Sub pixel locations and their interpolation complexities		
Sub Pixel Type	Points	Interpolation
(0, 0)	G	No
(0, 1/2), (1/2, 0)	b, h	1 6-tap
(0, 1/4), (1/4, 0), (0, 3/4), (3/4, 0)	a, c, d, n	1 6-tap + 1 2-tap
(1/4, 1/4), (1/4, 3/4), (3/4, 1/4), (3/4, 3/4)	e, g, p, r	2 6-tap + 1 2-tap
(1/2, 1/2)	j	7 6-tap
(1/2, 1/4), (1/4, 1/2), (3/4, 1/2), (1/2, 3/4)	i, f, k, q	7 6-tap + 1 Bilinear

[0038] It is clear that different interpolation methods have quite different computing complexities. Some up-to-date video codecs may even have more complex interpolation. For example, in recent 3D scalable video coding standards such as MC-EZBC, an 8-tap floating filtering process is used to achieve high interpolation accuracy.

[0039] Given the information about the interpolation cost associated with each type of motion vector, the basic premise behind reducing the decoder complexity is to select motion vectors that involve less interpolation complexity while maintaining high video quality. An empirical analysis of some H.264 statistical data shows that depending on the video content, 40% to 80% of motion vectors are located on sub pixels with different interpolation complexities. Therefore the principal approach to complexity reduction is to change motion vectors from high complexity sub pixel positions into low complexity, or even to integer-pixel positions.

[0040] In order to further reduce the temporal redundancy and improve the efficiency of motion estimation, H.264 defines a diverse set of block mode options. Besides the conventional modes (intra, forward, backward and bidirectional), two new important modes are introduced: variable block size and SKIP/DIRECT.

[0041] First, unlike earlier coding standards which use a fixed block size (usually 16×16 or 8×8) during motion estimation, H.264 allows for partitioning of an MB into several blocks with variable block size, ranging from 16 pixels to 4 pixels in each dimension. An MB can comprise up to 16 blocks. Each block with reduced size can have individual motion vectors to estimate the local motion at a finer granularity. Though such finer block sizes incur overhead such as

extra computation for searching and extra bits for coding the motion vectors, they allow more accurate prediction in the motion compensation process and consequently the residual errors can be considerably reduced (which is generally favorable for the final rate-distortion performance).

[0042] Secondly, the SKIP/DIRECT mode is utilized for the P/B frame in H.264 motion compensation to further increase the coding efficiency. The spatial/temporal neighbor motion vectors are used to predict the motion vector of the current block, without sending extra bits to encode the current motion vector. Details regarding the SKIP/DIRECT modes can be found in the Wiegand et al and Tourapis et al. references, cited and incorporated by reference hereinabove. In the mode decision algorithm used in accordance with the present invention, both the variable-size block mode and the SKIP/DIRECT mode may be considered during the search process.

[0043] The selection of block mode has a direct and substantial impact on the decoder computational complexity, because it determines what kind of motion vectors are recorded in the bit stream. Optimal selection of the block mode and the associated motion vectors are important problems addressed by the present invention. These steps are discussed in greater detail hereinafter.

[0044] As discussed above, conventional video coding systems generally encode a video bit stream by solving the optimization problem defined in Equation (1). The main control variables P involved in this procedure include motion vector v, block mode M and quantization parameter QP. There is complex interaction between the choices of these variables and thus the optimal solution is difficult to achieve. In practice, compromise approaches are taken and approximate solutions are often developed. For example, typically QP is determined through some empirical models and updated throughout the video sequence by some rate control algorithms. Given QP, the other variables, motion vector and block mode, are decided by applying some rate-distortion optimization process. A survey of these prior art procedures is described in G. J. Sullivan and T. Wiegand, Rate-Distortion Optimization for Video Compression IEEE Signal Processing Magazine, Vol. 15, Num. 6, pp. 74-90, Nov. 1998, which is incorporated herein by reference. A brief background summary follows.

[0045] Specifically, for each block B with a block mode M, the motion vector associated with the block is selected through a rate-distortion joint cost function:

$$V^*(B, M) = \underset{V \in \text{sup}\{V\}}{\text{argmin}} J_{MOTION}^{R,D} \quad (3)$$

$$(V | B, M) = \underset{V \in \text{sup}\{V\}}{\text{argmin}} \{D_{DFD}(V | B, M) + \lambda_{MOTION} R_{MOTION}(V | B, M)\}$$

where v^* is the optimal motion vector, $\text{sup}\{V\}$ defines the search space, whose dimensions include the prediction direction, the reference frame list and the search range. R_{MOTION} is the estimated bit rate to record the motion vector. D_{DFD} represents the prediction error between the current block and the reference block. Usually the sum of absolute difference (SAD) is adopted because the search space of motion vector is much larger than that of mode and SAD has lighter computation cost compared with the sum of squared difference (SSD). $J_{MOTION}^{R,D}(v)$ is the rate-distortion joint cost com-

prising of R_{MOTION} and $D_{DFD}\lambda_{MOTION}$ is the Lagrange multiplier to control the weight of the bit rate cost, relative to the signal distortion caused by the prediction error.

[0046] In a similar manner the block mode M for an MB is decided by the following.

$$\begin{aligned} M^*(MB, QP) &= \underset{M \in \text{sup}\{M\}}{\text{argmin}} J_{MODE}^{R,D} \\ (M | MB, QP) &= \underset{M \in \text{sup}\{M\}}{\text{argmin}} \{D_{REC}(M | MB, QP) + \lambda_{MODE} R_{REC} \\ &\quad (M | MB, QP)\} \end{aligned} \quad (4)$$

where M^* is the optimal block mode, and $\text{sup}\{M\}$ is the set of block mode options (such as INTRA, SKIP, DIRECT, FORWARD, BACKWARD, BIDIRECTION, etc). A full list of block mode options in H.264 can be found in the Tourapis et al. reference, cited and incorporated by reference hereinabove. D_{REC} is the SSD between the current MB and the reconstructed one through motion compensation. R_{REC} is the estimated bit rate associated with mode M . $J_{MODE}^{R,D}(M)$ is the joint cost comprising of rate R_M and distortion D_M , and λ_{MODE} is the Lagrange multiplier. The motion vectors associated with the optimal block mode $v^*(B, M^*)$ will be the final coded data recorded in the bit stream.

[0047] The Lagrange multipliers used in the above two cost functions determine the relative weights between signal quality and bit rate. To simplify the search process, an empirically derived relationship as the following is typically used in practice in the prior art. The square root relationship is partly due to the fact that SAD is used in modeling D_{DFD} while SSD is used for D_{REC} .

$$\lambda_{MOTION} = \sqrt{\lambda_{MODE}} \quad (5)$$

[0048] Rate control (RC) is the procedure of adjusting control variables so that the target rate requirement can be achieved while optimizing the overall video quality. Given a target bit rate, the average allocated bit rate may be computed for each basic coding unit. Then the Lagrange optimization method may be used to find the optimal set of control variables. However, searching over the entire variable space is very complex. In practice, most implementations use empirical models to restrict the search space. For example, a popular method, called rate-quantization modeling, maps the target bit rate to the quantization parameter, from which the Lagrange multipliers are decided. In addition, since coding of a data unit may not result in a bit rate that exactly matches the target, a separate process, called buffer management, may be used to monitor the available bit rate budget for the remaining data units and thus update the allocated recourse. A brief discussion of these processes is provided below.

[0049] A rate-Quantization (RQ) model describes the relationship between QP and the bit rate. A widely adopted quadratic RQ model, as described in T. Chiang and Y.-Q. Zhang, "A New Rate Control Scheme Using Quadratic Rate Distortion Model," IEEE Trans. Circuits Syst. Video Technol., Vol. 7, pp. 246-250, Feb. 1997, which is fully incorporated herein by reference, is:

$$R = D(P_1 \cdot QP^{-1} + P_2 \cdot QP^{-2}) \quad (6)$$

where D is the source complexity of the video signal, and usually measured using the motion estimation prediction errors (such as SAD), and $\{P_1, P_2\}$ are model parameters. Some systems use $P_2=0$ for simplicity. A typical RQ model-

ing procedure involves two major steps: model estimation and QP prediction. First several basic coding units are coded using some preset QP values. The coding units may include a certain number of MBs or one whole frame. The resulting rate-quantization-distortion (R-Q-D) points are collected. The model in Equation (6) is then estimated based on the observations. The estimated model can then be used to determine the QP value for the next coding unit based on the target bit rate R_1 and source complexity D_1 for the new unit. The former is determined by the buffer management process to be described below, and the latter is predicted using previous observations of the source complexity. Usually the source complexity is assumed to vary gradually and can be estimated using some simple relationship such as a linear model. Once coding of the new unit is completed, new observations of the R-Q-D points are collected and used to update the estimation of the RQ model in a sliding window manner. Namely, the oldest R-Q-D point is purged and the latest point is added to update the model.

[0050] The buffer management employs a virtual buffer to simulate the behavior of the data buffer on the decoder side. It is an important component in rate control in order to adjust the target bit rate for each coding unit and avoid the problem of buffer overflow or underflow. For example, given a target bit rate for the video sequence, the average bit rate allocation for each Group of Pictures (GOP) can be computed, and the allocated bit rate, R_1 , for a new frame to be coded (such as P frame) can be determined by monitoring the actual number of bits spent on the previous frames.

[0051] In H.264, given the target rate and QP for the coding unit, the following empirical relationship is often used to determine the Lagrange multiplier needed in the rate-distortion tradeoff optimization.

$$\lambda_{MODE} = 0.85 \times 2^{\frac{QP-12}{3}} \quad (7)$$

[0052] Such an empirical model is useful to simplify the search process in the Lagrange optimization method, while practical implementations have often shown satisfactory performance. Other parameters such as λ_{MOTION} can also be determined according to Equation (5).

[0053] The present invention is an improvement on the above procedures and provides a new system and method for complexity-adaptive motion estimation and mode decision ("CAMED"). In accordance with an exemplary embodiment of the present invention, given defined metrics for signal distortion and computational complexity, the tradeoff between video quality and resource consumption (both bit rate and computational complexity) may be considered to approximate the optimal motion vectors and block mode used in the motion compensation process in the decoder, thereby reducing decoding complexity and power consumption.

[0054] A system and method of the present invention may consist of several sub-components: a rate-distortion-complexity ("R-D-C") joint optimization framework, a complexity cost function, and a complexity control algorithm. The R-D-C framework extends the previously discussed Lagrange optimization framework to incorporate the complexity term. The complexity cost function provides quantitative measurements of the required computation for each

motion vector type. The complexity control algorithm is used to control the complexity over different coding units to meet the overall target complexity.

[0055] The present invention provides a solution to the problem defined in Equation (2), with an explicit Lagrange term to model the complexity cost. Therefore, in accordance with an exemplary embodiment of the present invention, the motion vectors may be selected through a rate-distortion-complexity joint cost function as follows:

$$V_C^*(B, M) = \underset{V \in \text{sup}(V)}{\text{argmin}} J_{MOTION}^{R,D,C} \quad (8)$$

$$(V | B, M) = \underset{V \in \text{sup}(V)}{\text{argmin}} \{J_{MOTION}^{R,D}(V | B, M) + \gamma_{MOTION} C_{MOTION}(V | B, M)\}$$

[0056] where C_{MOTION} is the complexity cost function associated with the selected motion vector ($V|B, M$), γ_{MOTION} is the Lagrange multiplier for the complexity term, $J_{MOTION}^{R,D}$ is the rate-distortion joint cost function defined in Equation (3), and $J_{MOTION}^{R,D,C}(v)$ is the rate-distortion-complexity joint cost function.

[0057] Similar to the earlier case described in Equation (4), the block mode search process of the present invention may be performed according to the following.

$$M_C^*(MB, QP) = \underset{M \in \text{sup}(M)}{\text{argmin}} J_{MODE}^{R,D,C} \quad (9)$$

$$(M | MB, QP) = \underset{M \in \text{sup}(M)}{\text{argmin}} \{J_{MODE}^{R,D}(M | MB, QP) + \gamma_{MODE} C_{MODE}(M | MB)\}$$

where C_{MODE} is the complexity cost function associated with the block mode, γ_{MODE} is the Lagrange multiplier, $J_{MODE}^{R,D}(v)$ is the rate-distortion joint cost function defined in (4), and $J_{MODE}^{R,D,C}(v)$ is the rate-distortion-complexity joint cost function.

[0058] The effect of $\gamma_{MODE}=0$ is apparent in view of the above. When $\gamma_{MODE}=0$, the solutions of Equations (8) and (9) are identical with the ones in Equations (3) and (4), namely no consideration is given to the complexity constraint and many motion vectors may be of sub-pixel values in order to minimize the distortion. When $\gamma_{MODE}=\infty$, all motion vectors are forced to integer pixel locations in order to minimize the complexity involved in interpolation for sub-pixel locations. Clearly there is a tradeoff between these two extremes to balance the performance in terms of quality and complexity.

[0059] For simplification, restrictions may be adopted, like those described in Equation (5), to limit the search space. For example, the following relationship may be used to link γ_{MODE} and γ_{MOTION} :

$$\gamma_{MOTION} = \sqrt{\gamma_{MODE}} \quad (10)$$

[0060] In the joint cost function described above, a quantitative model may be used to estimate the complexity associated with each candidate motion vector and block mode. As discussed above, the computational complexity is heavily influenced by the type of the motion vector (integer, half-pixel, or quarter-pixel) and the interpolation filters used in the motion compensation process. Focusing on the interpolation filtering cost, quantitative estimates of such complexities can be approximated by the number of filtering operations needed

in interpolation, such as those listed in Table 1. For example, using the same 6-tap filter and 2-tap filter implementations, the complexity of each motion vector type is as follows.

$$C_B(V) = N_B \cdot c_p(V) \quad (11)$$

$$c_p(V) = \begin{cases} 0 & V \text{ is integer } MV \\ e_6 & V \text{ is subpixel } b, h \\ e_6 + e_2 & V \text{ is subpixel } a, c, d, n \\ 2e_6 + e_2 & V \text{ is subpixel } e, g, p, r \\ 7e_6 & V \text{ is subpixel } j \\ 7e_6 + e_2 & V \text{ is subpixel } i, f, k, q \end{cases} \quad (12)$$

where $c_B(v)$ is the computational cost for the current coding block, V is the motion vector, $c_p(V)$ is the computational complexity required for calculating a reference pixel, N_B is the number of pixels in the current coding block, and $\{e_6, e_2\}$ are the estimated complexities for 6-tap and 2-tap interpolation respectively. Experimental results indicate that a simplified model ignoring the 2-tap interpolation will mostly result in the same selection of the motion vectors. With such simplification, the above model may be reduced to the following with a common factor e_6 removed.

$$c_p(V) = \begin{cases} 0 & \text{integer } MV \\ 1 & \text{subpixel } a, b, c, d, h \text{ \& } n \\ 2 & \text{subpixel } e, g, p, r \\ 7 & \text{subpixel } i, j, f, k, q \end{cases} \quad (13)$$

[0061] Equations (11) and (13) may be used to estimate the computational complexity based on the interpolation operation—these models provide platform-independent modeling. Alternatively, the complexity cost may be derived from specific software or hardware implementations (platform-dependent modeling). The following are two examples of platform-dependent modeling in accordance with exemplary embodiments of the present invention.

[0062] The complexity cost functions defined in Equation (11) and (13) are also considered to be pixel-based in that the complexity is calculated for each pixel independently without considering the reusability of previous calculated pixel (or sub pixel) values. For block-based motion compensation as adopted in H.264, some interpolations can be saved by directly using previous computed results. Again according to the H.264 standard description, the following categories of sub pixels may be considered:

[0063] For integer pixel no interpolation is necessary, and the complexity is zero;

[0064] For sub pixels a, b, c, d, h and n, they are located in either integer row or integer column, only one 6-tap filtering is necessary for them.

[0065] Considering a 4×4 block (the minimum MC unit in H.264), the complexity is 1×16=16;

[0066] For sub pixels e, g, p and r, similar to previous case, the complexity is 2×16=32;

[0067] For sub pixels i, j, f k and q, on each column within a 4×4 block, the topmost sub pixel requires full 7 6-tap interpolations. Whereas for each of the remaining three sub pixels located in the same column, 5 6-tap interpolations calculating its upper sub pixel value can

be reused and only two additional 6-tap interpolations are necessary. Therefore, the complexity is $7 \times 4 + 2 \times 12 = 52$.

[0068] Therefore, block-based complexity may be modeled as (after value scaling):

$$C_p(V) = \begin{cases} 0 & V \text{ is integer } MV \\ 4 & V \text{ is subpixel } a, b, c, d, h \text{ and } n \\ 8 & V \text{ is subpixel } e, g, p, r \\ 13 & V \text{ is subpixel } i, j, f, k, q \end{cases} \quad (14)$$

[0069] The model in Equation (14) above can even be further fine tuned considering variable block size implementation during MC, in accordance with the lookup table below:

TABLE 2

Lookup table for complexity cost using variable block size MC implementation				
Mode	Integer	1 6-tap	2 6-tap	7 6-tap
SKIP/DIRECT	0	256	512	592
16 × 16	0	256	512	592
16 × 8	0	128	256	296
8 × 16	0	128	256	296
8 × 8	0	64	128	168
8 × 4	0	32	64	84
4 × 8	0	32	64	84
4 × 4	0	16	32	52
Intra	0	0	0	0

[0070] In an exemplary hardware implementation of an embodiment of the present invention, each interpolation operation can be divided into a number of basic operators such as addition, shifts, and/or multiplications. In this case, $\{e_6, e_2\}$ can be modeled with more detail, such as:

$$e_i = \sum_j \rho N(o_j) P(o_j), i = 2, 6 \quad (15)$$

where o_j is the basic operator involved in the interpolation implementation, $N(o_j)$ is the required number of operator o_j , $P(o_j)$ is the power consumption of operator o_j , and $\rho \geq 1$ is the adjustment factor to consider additional power cost such as memory access. For instance, a hardware implementation of interpolation was introduced in T.-C. Chen, Y.-C. Huang and L.-G. Chen, "Full Utilized and Resuable Architecture for Fractional Motion Estimation of H.264/AVC", ICASSP2004, Montreal, Canada, May 17-21, 2004, which is fully incorporated herein by reference. Its estimated complexity is

$$e_6 = P(6P_{add} + 2P_{shift}) \quad (16)$$

where P_{add} , P_{shift} are the power consumption for the addition operator and the 2-bit shift operator respectively.

[0071] Each block may be associated with multiple reference blocks, each of which needs a motion vector. For example, for bidirectional prediction, each block may need two motion vectors for forward and backward prediction respectively. Thus, the computational cost for a block B with the block mode M is calculated as:

$$C_{MOTION}(V | B, M) = \sum_j (c_B(V_j, M, B)) \quad (17)$$

where the summation is over each reference block.

[0072] Each MB may consist of several smaller blocks, depending on the block mode, M. The overall computational cost associated with a MB and a block mode can be calculated as:

$$C_{MODE}(M | MB) = \sum_i \sum_j (c_B(B_i, V_j, MB)) \quad (18)$$

where i is the index of the individual blocks contained in the MB, and j is the index for multiple motion vectors associated with a single block. Equations (17) and (18) are generic and applicable to all inter-coded block modes, including forward/backward/bi-directional motion compensation and SKIP/DIRECT.

[0073] Equations (8) and (9) use the Lagrange multiplier to formulate R-D-C combined optimization problems. Assuming that the selection of motion vector and block mode for a certain MB is independent of behaviors in other MBs (which is a reasonable approximation of the real case), at the optimal solution each MB will have the same Lagrange multiplier (γ_{MOTION} , γ_{MODE}). This is an important property of the Lagrange multiplier. In other words, given a specific $\hat{\gamma}_{MODE}$ and considering Equation (10), the bit stream with complexity $C(\hat{\gamma}_{MODE})$ can be derived. This $\hat{\gamma}_{MODE}$ is (approximately) the optimal solution for the following problem:

$$\min_{\{V, M\}} \sum_{i=1}^N J^{R,D}(V_i, M_i) \text{ s.t., } \sum_{i=1}^N C(V_i, M_i) \leq C(\hat{\gamma}_{MODE}) \quad (19)$$

[0074] where $\vec{V} = (V_1, V_2, \dots, V_N)$ and $\vec{M} = (M_1, M_2, \dots, M_N)$ are the motion vectors and block modes for all MBs respectively, V_i and M_i are the motion vector and block mode for i^{th} MB respectively, $J^{R,D}$ is the R-D cost function, and $C(V, M)$ is the complexity cost function. Unfortunately, the complexity level $C(\hat{\gamma}_{MODE})$ associated with $\hat{\gamma}_{MODE}$ cannot be known in advance unless the bit stream has been encoded. Therefore, the Lagrange multiplier has to be adjusted in order to match certain target complexity level. This procedure implemented in the context of the present invention is referred to as "complexity control."

[0075] Complexity control, analogous to the rate control process described above, is a process to allocate the complexity resource among the coding units and to determine parameters like Lagrange multiplier γ_{MODE} to be used in the optimization procedure. In the exemplary embodiment of the present invention described above, the allocated bit rate is mapped to the quantization parameter, which in turn is used to find the Lagrange multiplier λ_{MODE} . In the following exemplary embodiment of the present invention, two components of the complexity control algorithm are described—complexity modeling and buffer management. The former is used to characterize the relationship between the target complexity and the Lagrange multiplier γ_{MODE} . The latter is for monitor-

ing the complexity usage and updating the available computational resource for each new data unit.

[0076] In complexity control a feasible modeling of complexity and control parameters (γ_{MODE} in this case) is necessary. One of the objectives of this modeling is to identify the relationship between the target complexity and the optimization control parameter, γ_{MODE} . Based on experimentation, there is an approximately linear relationship between the complexity value and log of the Lagrange multiplier, and the type of the frame (B or P) greatly influences this relationship. [0077] A reasonable model based on these observations is as follows:

$$C(\gamma_{MODE}) = D(K_1 \ln(\gamma_{MODE}) + K_0) \quad (20)$$

where C is the complexity, D is a factor measuring the video source complexity similar to that used in Equation (6) for rate control. K_1 , K_0 are the model parameters obtained during the coding procedure. Due to different coding mechanism, P and B frames will have distinguished model parameters and may need to be handled separately.

[0078] The above model is driven by empirical simulation observations. The linear dependence of the computational complexity on the signal source complexity is also intuitive—the more complex the signal source is, the higher the accuracy required to estimate the motion vector, and thus there is a larger gain in using sub-pixel motion vectors, resulting in an increased computational cost.

[0079] Using this model, the Lagrange multiplier $\gamma_{MODE}(t)$ for the current coding unit t can be determined by the following:

$$\gamma_{MODE}(t) = \exp\left\{\frac{C(t) - K_0 D(t)}{K_1 D(t)}\right\} \quad (21)$$

where C(t) is the allocated computational budget and D(t) is the predicted complexity measurement for unit t. In practice, in order to avoid large quality fluctuation, the change rate of $\gamma_{MODE}(t)$ may be bounded by some thresholds.

[0080] Referring now to FIG. 3, a flow chart for a method for optimizing the selection of motion vectors and motion compensation block modules in a video decoder in accordance with the above-described present invention is shown. In step 310, an input video data sequence is received. Next, in step 320, a target complexity level for the video decoder is determined. Then, in step 330, a Lagrange multiplier is determined. Next, in step 340, for each macroblock, a motion vector is calculated for one or more of the H.264 block modes. This may preferably be performed for every block mode available in H.264 (or the applicable video standard). Next, in step 350, for each macroblock, a best block mode is selected, based on the Lagrange multiplier. Finally, in step 360, a complexity-control algorithm, as described in detail above, is applied to encode the received input video data sequence in accordance with the identified target complexity level, such that, upon decoding, a consistent complexity level is maintained throughout decoding.

[0081] Referring now to FIG. 4, a flow chart for another method in accordance with the above-described present invention for selecting motion vectors in an optimized video decoder is shown. In step 410, an input video data sequence comprising one or more macroblocks is received. In step 420, for each macroblock, at least one inter-predictive block mode is enumerated. In step 430, for each block of each enumerated

block mode, one or more motion vectors that yields the minimum rate-distortion-complexity is selected. Finally, in step 440, the selected one or more motion vectors is stored.

[0082] Referring now to FIG. 5, a flow chart for another method in accordance with the above-described present invention for selecting a block mode in an optimized video decoder is shown. In step 510, an input video data sequence is received. Next, in step 520, one or more possible block modes for each macroblock is selected. In step 530, one or more motion vectors for each block mode is received. A rate-distortion-complexity cost for each block mode is then calculated in step 540. In step 550, for each macroblock, a block mode is selected that yields a minimum rate-distortion-complexity cost function. Finally, in step 560, the selected block mode is stored for further processing.

[0083] In accordance with an exemplary embodiment of the present invention, a complexity buffer may also be implemented. A complexity buffer is a virtual buffer used to simulate the complexity usage status on the decoder side. It is analogous to the rate buffer used in the rate control to update the estimation of available resources and avoid issues of buffer overflow or underflow. Denoting CGOP the remaining complexity budget in one GOP, N_P , N_B the remaining numbers of P, B frames respectively, and η the complexity ratio between P and B, which is updated during video coding, the target complexity levels for P, B frame C_P , C_B may be calculated using the following equations:

$$\frac{C_B}{C_P} = \eta \quad (22)$$

$$N_P C_P + N_B C_B = C_{GOP} \quad (23)$$

[0084] Once C_P , C_B are available, $\gamma_{MODE}(t)$ is determined using the models and techniques described above. The formulations in Equations (22) and (23) assume the basic coding unit as one frame. It can be easily extended to smaller units for a finer granularity.

[0085] Notably, experiments involving an exemplary implementation of the present invention using an H.264 codec over different video sequences, different bit rates, and different complexity levels demonstrated that up to 60% of the interpolation complexity can be saved at the decoder without incurring noticeable quality loss (within 0.2 dB). Even for challenging video clips, 33% of the complexity can be reduced with quality difference less than 0.3 dB. Accordingly, the present invention can improve video decoding systems to reliably meet target complexity requirements for a wide range of video content.

[0086] As noted previously, the exemplary system as described herein is not limited to H.264, but may be implemented in accordance with any video encoding/decoding system, including systems operating in accordance with the MPEG-4 standard, the Motion Compensated Embedded Zero Block Coding ("MC-EZBC") standard, and others. Additionally, the present invention may be implemented using software, hardware, or some combination thereof, as would be understood by one of ordinary skill in the art. The scope of the invention is not limited to any particular implementation of the inventive system and method described herein.

[0087] The foregoing merely illustrates the principles of the invention. Various modifications and alterations to the described embodiments will be apparent to those skilled in

the art in view of the teachings herein. It will thus be appreciated that those skilled in the art will be able to devise numerous systems and methods which, although not explicitly shown or described herein, embody the principles of the invention and are thus within the spirit and scope of the invention.

REFERENCES

- [0088] [1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, "Overview of the H.264/AVC Video Coding Standard," IEEE Trans. Circuits Syst. Video Technol., vol 13, pp. 560-576, Jul. 2003.
- [0089] [2] T. Wedi; H. G. Musmann, Motion- and aliasing-compensated prediction for hybrid video coding, IEEE Trans. Circuits Syst. Video Technol., vol 13, pp. 577-586, Jul. 2003.
- [0090] [3] V. Lappalainen, A. Hallapuro, and T. D. Hämäläinen, "Complexity of Optimized H.26L Video Decoder Implementation," IEEE Trans. Circuits Syst. Video Technol., vol 13, pp. 717-725, Jul. 2003.
- [0091] [4] A. M. Tourapis, F. Wu, S. Li, "Direct mode coding for bi-predictive pictures in the JVT standard," ISCAS2003, vol. 2, 700-703, Thailand, 2003.
- [0092] [5] G. J. Sullivan and T. Wiegand, Rate-Distortion Optimization for Video Compression IEEE Signal Processing Magazine, Vol. 15, Num. 6, pp. 74-90, Nov. 1998.
- [0093] [6] T. Chiang and Y.-Q. Zhang, "A New Rate Control Scheme Using Quadratic Rate Distortion Model," IEEE Trans. Circuits Syst. Video Technol., Vol. 7, pp. 246-250, Feb. 1997.
- [0094] [7] T.-C. Chen, Y.-C. Huang and L.-G. Chen, "Full Utilized and Resuable Architecture for Fractional Motion Estimation of H.264/AVC", ICASSP2004, Montreal, Canada, May 17-21, 2004.
- [0095] [8] X. Zhou, E. Li, and Y.-K. Chen, "Implementation of H.264 Decoder on General-Purpose Processors with Media Instructions", in Proc. of SPIE Visual Communications and Image Processing, Jan. 2003.
- [0096] [9] National's PowerWise™ technology. <http://www.national.com/appinfo/power/powerwise.html>
- [0097] [10] Y. Eisenberg, C. E. Luna, T. N. Pappas, R. Berry, A. K. Katsaggelos, Joint source coding and transmission power management for energy efficient wireless video communications, CirSysVideo(12), No. 6, June 2002, pp. 411-424.
- [0098] [11] Q. Zhang, W. Zhu, Zu Ji, and Y. Zhang, "A Power-Optimized Joint Source Channel Coding for Scalable Video Streaming over Wireless Channel", IEEE International Symposium on Circuits and Systems (ISCAS) 2001, May, 2001, Sydney, Australia.
- [0099] [12] X. Lu, E. Erkip, Y. Wang and D. Goodman, "Power efficient multimedia communication over wireless channels", IEEE Journal on Selected Areas on Communications, Special Issue on Recent Advances in Wireless Multimedia, Vol. 21, No. 10, pp. 1738-1751, Dec., 2003.
- [0100] [13] H. Kim and Y. Altunbasak, "Low-complexity macroblock mode selection for the H.264/AVC encoders," IEEE Int. Conf. on Image Processing, Suntec City, Singapore, October 2004.
- [0101] [14] A. Ray and H. Radha, "Complexity-Distortion Analysis of H.264/JVT Decoder on Mobile Devices," Picture Coding Symposium (PCS), December 2004.
- [0102] [15] K. Lengwehasatit and A. Ortega, "Rate Complexity Distortion Optimization for Quadtree-Based DCT Coding", ICIP 2000, Vancouver, BC, Canada, September 2000.
- [0103] [16] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-Rate-Distortion Analysis for Wireless Video Communication under Energy Constraints," IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on Integrated Multimedia Platforms, 2004.
- [0104] [17] B. Girod, A. Aaron, S. Rane and D. Rebollo-Monedero, "Distributed video coding," *Proc. of the IEEE*, Special Issue on Video Coding and Delivery, 2005.
- [0105] [18] A. M. Tourapis. "Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation," Proceedings of Visual Communications and Image Processing 2002 (VCIP-2002), San Jose, Calif., Jan 2002, pp. 1069-79.
- [0106] [19] H.-Y. Cheong, A. M. Tourapis, "Fast Motion Estimation within the H.264 codec," in proceedings of ICME-2003, Baltimore, Md., Jul. 6-9, 2003.
- [0107] [20] M. Schaar, H. Radha, Adaptive motion-compensation fine-granular-scalability (AMC-FGS) for wireless video, IEEE Trans. on CSVT, vol, 12, no. 6, 360-371, 2002.
1. A method for optimizing the selection of motion vectors and motion compensation block modules in a video encoder comprising:
 - receiving an input video data sequence comprising at least one macroblock;
 - identifying a target complexity level for the video data sequence;
 - determining a Lagrange multiplier for the video data sequence;
 - for each at least one macroblock, calculating at least one motion vector for one or more block modes based on the determined Lagrange multiplier;
 - for each at least one macroblock, selecting one of the one or more block modes based on the determined Lagrange multiplier;
 - encoding the received input video data sequence in accordance with the identified target complexity level to produce an encoded bitstream that may be decoded with consistent complexity.
 2. The method of claim 1, wherein the step of encoding the received input video data sequence comprises applying a complexity control algorithm.
 3. The method of claim 1, wherein the received input video data is data in accordance with the H.264 standard.
 4. The method of claim 1, wherein the received input video data is data in accordance with the MPEG-4 standard.
 5. The method of claim 1, wherein the received input video data is data in accordance with the Motion Compensated Embedded Zero Block Coding (MC-EZBC) standard.
 6. A method for selecting motion vectors for optimized video encoding comprising:
 - receiving an input video data sequence comprising one or more macroblocks;
 - for each macroblock, enumerating at least one inter-predictive block mode whose one or more motion vectors are to be calculated based on motion estimation;
 - for each block of each enumerated block mode, selecting the one or more motion vectors that yield minimum rate-distortion-complexity;
 - storing the selected one or more motion vectors for each block of each enumerated block mode.

7. The method of claim 6, wherein the received video frame data is data in accordance with the H.264 standard.

8. The method of claim 6, wherein the received video frame data is data in accordance with the MPEG-4 standard.

9. The method of claim 6, wherein the received video frame data is data in accordance with the Motion Compensated Embedded Zero Block Coding (MC-EZBC) standard.

10. A method for selecting a block mode for optimized video encoding comprising:

receiving an input video data sequence comprising one or more macroblocks;

identifying one or more possible block modes for each macroblock;

retrieving one or more selected motion vectors for each block mode;

using the selected motion vectors, calculating the rate-distortion-complexity cost for each block mode;

selecting, for each macroblock, a block mode that yields a minimum rate-distortion-complexity cost function;

storing at least one of the selected block modes for further processing.

11. The method of claim 10, wherein the received video frame data is data in accordance with the H.264 standard.

12. The method of claim 10, wherein the received video frame data is data in accordance with the MPEG-4 standard.

13. The method of claim 10, wherein the received video frame data is data in accordance with the Motion Compensated Embedded Zero Block Coding (MC-EZBC) standard.

14. An optimized video encoding system comprising:

at least one processor;

a memory coupled to the at least one processor and containing instructions which, when executed by the processor, cause the processor to perform the steps of:

receiving an input video data sequence comprising at least one macroblock;

identifying a target complexity level for the video data sequence;

determining a Lagrange multiplier for the video data sequence;

for each at least one macroblock, calculating at least one motion vector for one or more block modes based on the determined Lagrange multiplier;

for each at least one macroblock, selecting one of the one or more block modes based on the determined Lagrange multiplier;

encoding the received input video data sequence in accordance with the identified target complexity level to produce an encoded bitstream that may be decoded with consistent complexity.

15. The method of claim 14, wherein the memory further contains instructions for applying a complexity control algorithm in conjunction with the encoding of the received input video data sequence.

16. The system of claim 14, wherein the received input video data sequence is data in accordance with the H.264 standard.

17. The system of claim 14, wherein the received input video data sequence is data in accordance with the MPEG-4 standard.

18. The system of claim 14, wherein the received input video data sequence is data in accordance with the Motion Compensated Embedded Zero Block Coding (MC-EZBC) standard.

19. An optimized video encoding system comprising:

at least one processor;

a memory coupled to the at least one processor and containing instructions which, when executed by the processor, cause the processor to perform the steps of:

receiving an input video data sequence comprising one or more macroblocks;

identifying one or more possible block modes for each macroblock;

retrieving one or more selected motion vectors for each block mode;

using the selected motion vectors, calculating the rate-distortion-complexity cost for each block mode;

selecting, for each macroblock, a block mode that yields a minimum rate-distortion-complexity cost function;

storing at least one of the selected block modes for further processing.

20. An optimized video encoding system comprising:

at least one processor;

a memory coupled to the at least one processor and containing instructions which, when executed by the processor, cause the processor to perform the steps of:

receiving an input video data sequence comprising one or more macroblocks;

for each macroblock, enumerating at least one inter-predictive block mode whose one or more motion vectors are to be calculated based on motion estimation;

for each block of each enumerated block mode, selecting the one or more motion vectors that yield minimum rate-distortion-complexity;

storing the selected one or more motion vectors for each block of each enumerated block mode.

* * * * *