Journal of King Saud University – Computer and Information Sciences (2018) 30, 249–258

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Adaptive order search and tangent-weighted trade-off for motion estimation in H.264

Srinivas Bachu [a,*], K. Manjunath Achari [b]

[a] Department of ECE, GITAM University, Hyderabad, Telangana, India
[b] Department ECE, GITAM University, Hyderabad, Telangana, India

Received 27 March 2016; revised 21 July 2016; accepted 26 July 2016
Available online 2 August 2016

KEYWORDS

Block motion estimation;
Square search;
Hexagon search;
H.264;
Video coding

Abstract Motion estimation and compensation play a major role in video compression to reduce the temporal redundancies of the input videos. A variety of block search patterns have been developed for matching the blocks with reduced computational complexity, without affecting the visual quality. In this paper, block motion estimation is achieved through integrating the square as well as the hexagonal search patterns with adaptive order. The proposed algorithm is called, AOSH (Adaptive Order Square Hexagonal Search) algorithm, and it finds the best matching block with a reduced number of search points. The searching function is formulated as a trade-off criterion here. Hence, the tangent-weighted function is newly developed to evaluate the matching point. The proposed AOSH search algorithm and the tangent-weighted trade-off criterion are effectively applied to the block estimation process to enhance the visual quality and the compression performance. The proposed method is validated using three videos namely, football, garden and tennis. The quantitative performance of the proposed method and the existing methods is analysed using the Structural SImilarity Index (SSIM) and the Peak Signal to Noise Ratio (PSNR). The results prove that the proposed method offers good visual quality than the existing methods.

1. Introduction

The advancements in mobile communication technologies have enabled the portable devices to execute complex multimedia applications, which involve video processing. The development in multimedia services has facilitated the crucial role of video compression in reducing the bandwidth, which is required for the transmission as well as the storage of information in various applications (Chen et al., 2012; Ma and Kuo, 2007). The ideo combines a sequence of images or frames, along the temporal and the spatial dimensions. Two successive frames may be approximately similar, but the difference in the frames may be due to the change in the position of the object or the camera or due to noise. The quantity of data, which is used for representing the video frames, can be reduced by video compression. In order to compress the data that are to be transmitted, it is important to determine the redundancy of the data for their use in defining the predictable properties.

* Corresponding author.
E-mail address: srinivasbachu14@gmail.com (S. Bachu).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2016.07.002

These properties are used to predict the real data that are to be transmitted. But, the errors between the predicted and the real data are only transmitted (Fabrizio et al., 2012).

There are several algorithms for video compression. Among them, H.264 is a standard video coding algorithm (Wiegand et al., 2003; ITU-T and ISO/IEC JTC 1, 2010). While comparing with the existing algorithms (ITU-T Rec. H.261, 1993; ITU-T Rec. H.263, 1995), H.264 provides better performance using a set of advanced coding tools such as, the spatial-domain intra prediction, the variable block-size Motion Estimation (ME) (Pan and Kwong., 2011; Pan et al., 2012), a Motion Vector (MV) with quarter pixel accuracy, the multiple reference frames, a de-blocking filter, the Rate-Distortion (R-D) optimization Sullivan and Wiegand, 1998; Wang and Kwong, 2008 and an entropy coding. The Motion Estimation method involves a computationally complex process, as it uses blocks of varying sizes and multiple reference frames. In this method, each frame is divided into blocks and predicted using the correct match from the reference frame. The assumption behind here is that each block is subjected to the independent motion from the reference block. The performance of the matching process depends on factors such as, the block size and shape, matching precision and the motion model.

Different research efforts are dedicated to represent these problems. Accordingly, H.264 (ITU-T Rec.H.264, 2003) has used an advanced tree-structured block portioning method that partitions a frame into a block of size 4×4. Several problems occur in this standard because this standard makes use of multiple reference frames, several motion estimation modes and different sub-blocks. So, the computational complexity of the block motion estimation process in H.264 is very high than that of other standards. The full search algorithm (Dufaux and Moscheni, 1995) for the block motion estimation process checks each candidate motion vector. It is a brute-force algorithm that has high computational complexity and high accuracy. So, the researchers are now trying to find a block motion estimation algorithm with low complexity and high accuracy (Duanmu and Yu, 2012). Some researchers have used motion-assisted merging or leaf-merging (Wang et al., 2004), after quad tree-based segmentation (Mathew and Taubman, 2006), to reduce the number of segments or blocks that can be predicted using the same motion vector. In Huang et al. (2006) and Zhao et al. (2010), the motion estimation process has been proved to consume an encoding time that range from 70% (one reference frame) to 90% (five reference frames) of the total encoding time of a standard H.264/AVC encoder.

In this paper, we develop an adaptive order search and tangent-weighted trade off for achieving motion estimation in the H.264 coding standard. At first, the input video is read out and the frames are extracted. These frames are then divided to form a set of blocks. The proposed hybrid search algorithm, called AOSH algorithm, is applied on these blocks of the frames to disclose the best matching block. The proposed search algorithm integrates the square as well as the hexagon search with the adaptive order based on the rate-distortion trade off to find the best matching point with minimum searching points. The tangent-weighted trade-off is newly developed to evaluate the matching point. This process considers the rate and the distortion parameters, which are then weighted by a tangent function. Once a suitable search point is selected, the motion vectors are identified and applied with the integer transformation and quantization process. Finally, CAVLC (Context-Adaptive Variable Length Coding) is applied to convert the quantized values into a bit stream.

The main contributions of this paper are as follows:

- A new search algorithm, called AOSH algorithm, is newly developed through integrating the square search and the hexagonal search. Here, the order of the search is adaptively changed to find the matching point with large ease and less searching time.
- Tangent-weighted trade-off is newly developed by considering two objective parameters, namely, the rate and the distortion. In the tangent-weighted trade-off criterion, the rate and the distortion parameters are weighted using the tangent function.

This paper is organized as follows: Section 2 reviews the literature and the motivating scenario. Section 3 presents the proposed adaptive order search and the tangent-weighted trade-off for H.264. Section 4 discusses about the experimentation and the evaluation of the proposed method. Finally, the conclusion is given in Section 5.

2. Literature review

Table 1 reviews few recent works, which have aimed to accomplish motion estimation for video compression. Chen et al. (2012) have presented a lossless video compression system based on a Backward Adaptive pixel-based fast Predictive Motion Estimation (BAPME). Unlike the widely used block-matching motion estimation techniques, this scheme predicted

Table 1 Literature review.

Author	Contributions	Issues
Chen et al. (2012)	Adaptive pixel-based motion estimation	Refines search classifications that affect the motion vectors severely
Liu (2014)	Unsymmetrical-cross multi-hexagon grid search-based motion estimation	Requires more computational effort to search the partitions
Pan·and Kwong (2013)	Direction-based unsymmetrical cross multi-hexagon grid search-based motion estimation	Recursive search process takes too long time to find the search patterns
Muhit et al. (2010)	Elastic motion model with larger blocks	Small level motion preservation is tough with larger blocks
Muhit et al. (2012)	Geometry-adaptive partitioning-based elastic motion model	Limited number of searches over the partitions
Duanmu and Zhang (2012)	Octagon and Triangle search-based motion estimation	Finds difficulty in handling tricky motions such as, zoom and rotation
Bosch et al. (2011)	Visual motion perception-based motion model	Chance of missing the small level motions due to the visual motion perception model
Fabrizio (2012)	Tangent distance prediction-based motion compensation	Fails to preserve the motion discontinuities

the motion on a pixel-by-pixel basis by comparing a group of past observed pixels between two adjacent frames, eliminating the need of transmitting side information. Liu et al., 2014 have proposed an adaptive motion estimation scheme to further reduce the calculation redundancy of UMHexagonS. Firstly, motion estimation search patterns have been designed according to the statistical results of motion vector (MV) distribution information. Then, a MV distribution prediction method is designed, including prediction of the size of MV and the direction of MV. At last, according to the MV distribution prediction results, achieve self-adaptive subregional searching by the estimation search patterns. Pan·and Kwong, (2013) have developed a Direction-based UMHexagonS (DBUMHexagonS) to further reduce the computational complexity of UMHexagonS. Each block matching search pattern of UMHexagonS was divided into four direction-based subsearch patterns, one of four directions was selected according to the difference between the Motion Vector (MV) of current block and the MV of its collocated block in previous frame, such a direction was applied to all following search patterns.

Muhit et al., (2010) have proposed an extended prediction strategy that incorporates non-translational motion prediction. This method used an elastic motion model with 2-D cosine basis functions to estimate non-translational motion between the blocks. To achieve superior performance, the scheme takes advantage of larger blocks with multi-level partitioning. Muhit et al. (2012) have developed an efficient block-partitioning scheme that incorporates both geometry-adaptive partitioning and an elastic motion model as extensions to the standard motion estimation procedure. Duanmu and Yu (2012) have presented a fast block motion estimation algorithm, which reduces the computational complexity of the UMHexagonS algorithm by more than 33 percent and have about the same accuracy. This was achieved by first classifying the motion type of a block. Bosch et al. (2011) have described several spatial-texture models for video coding. They investigated several texture features in combination with two segmentation strategies in order to detect texture regions in a video sequence. These detected areas were not encoded using motion compensated coding. The model parameters were sent to the decoder as side information. After the decoding process, frame reconstruction was done by inserting the skipped texture areas into the decoded frames. Fabrizio et al., 2012 have described several spatial-texture models for video coding. They investigated several texture features in combination with two segmentation strategies in order to detect texture regions in a video sequence. These detected areas were not encoded using motion compensated coding. The model parameters were sent to the decoder as side information. After the decoding process, frame reconstruction was done by inserting the skipped texture areas into the decoded frames.

2.1. Motivating scenario

Due to the significant application of H.264 or MPEG in video compression, various researchers have tried to develop different variants of the video compression model based on this standard. Specifically, motion estimation and the finding of the motion vector through different search algorithms have been researched majorly. The literature (Chen et al., 2012; Liu et al., 2014; Pan and Kwong, 2013; Muhit et al., 2010, 2012; Duanmu and Yu, 2012; Bosch et al., 2011; Fabrizio et al., 2012) presents different algorithms for motion search and estimation. These works are mostly concentrated towards developing search algorithms that enable the best matching block to be easily found, in order to reduce the computation overhead and to enhance the visual quality as well as the compression performance. The search algorithm mostly requires much computational time to find the motion blocks. The best matching block should satisfy the two constraints that are defined in the trade-off criterion. The matching of the block in the current frame and the reference frame starts up with the square search, as most of the videos are symmetrical in nature. However, the square search finds difficulty in preserving the motion discontinuities, in case of the difficult motions like, zooming, rotation, fast moving objects and so on.

In order to preserve these discontinuities, hexagonal search and octagonal search is introduced. These searches allow the best matching block to be easily found by covering the edge points of the search area. This fact clearly denotes that the shape and the size of the search patterns have a major impact on the video quality (error performance), in addition to the computational complexity of the motion estimation procedure. The intrinsic goal here is to find the matching point that is similar to the reference frame. If more search points are used to find the best matching point, the complexity would be high. But, the increase in the search points will cause a greater gain in video compression. Hence, finding the better matching point to estimate the motion field is a reasonable progression in the video coding research. In the searching algorithm, the selection of the better search points is always evaluated by an objective function. Here, the rate and the distortion parameters are taken to evaluate the search points. In this paper, the design of the rate-distortion trade-off plays a lead role in selecting the blocks for further partitioning or encoding.

3. Proposed adaptive order search and tangent-weighted trade-off for motion estimation in H.264

This section presents the proposed adaptive order search and tangent-weighted trade-off for achieving motion estimation in H.264. At first, the input video is read out and the frames are extracted. Then, these frames are split into macro blocks for performing the hybrid search. The proposed hybrid search algorithm, called AOSH algorithm, integrates the square as well as the hexagonal search with the higher order using the rate-distortion trade-off. The searching can be accomplished either by hexagon or square and the required order is based on the characteristic of the proposed rate-distortion trade-offs. Once the suitable search point is selected and after performing the processes like, motion vector computation, integer transformation and quantization, the encoding process is performed using CAVLC. Fig. 1 shows the block diagram of the proposed adaptive order search and tangent-weighted trade-off for estimating motion in H.264.

3.1. Adaptive order square search

Square search is one of the important search methods, which is modified during the process of adaptive order square search. Here, the square search and the order are adaptively selected based on the trade-off criterion. In the square search for

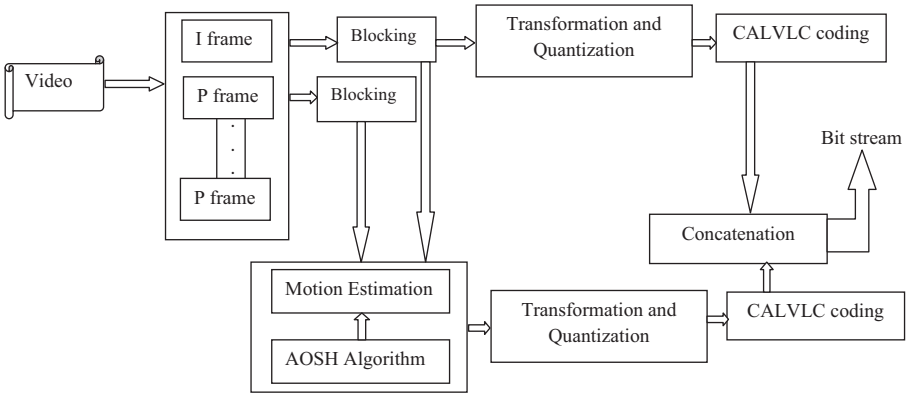

Figure 1 Block diagram of the proposed adaptive order search and tangent-weighted trade-off for Motion Estimation in H.264.

motion estimation, the symmetric pattern is generated initially by locating four points in the corners and one point in the centre to form a square pattern. Hence, the total number of search points that is visited by the square search will be five. Since the initial search pattern includes all the directions, the square search can achieve quick as well as easier detection of any motion and its associated direction.

The usual motion is mostly symmetrical, either in the downward or the upward direction. Hence, the square search is beneficial, robust and easy to implement in hardware. The symmetrical searching capability of the square search aids well in the detection of the best matching block, if there is no abnormal motion behaviour in the video. In the adaptive order square search algorithm, the movement of the search point is obtained through the search of five points with an order of either 0 or 1. Order is a parameter, which is introduced to handle the size of the square that enables searching. Using the order and the constant parameter, the depth of square can be easily found out as:

$$d = O + C \tag{1}$$

where, O indicates the order of the search, d denotes the depth or size of the square. The size of the square depends on the order of the search and a search parameter C. C is a constant that is to be fixed manually. Due to the nature of the order value, which is either 0 or 1, two sets of square points are introduced here to search and find the best matching block.

Assume (x, y) as the centre location of the search point. The searching of the best matching block using the square search is possibly done with the following four points, along with the centre points.

$$SP_1^S = (x+p, y+q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{2}$$

$$SP_2^S = (x-p, y+q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{3}$$

$$SP_3^S = (x+p, y-q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{4}$$

$$SP_4^S = (x-p, y-q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{5}$$

where, SP_1^S, SP_2^S, SP_3^S and SP_4^S are the search points that are defined in the square region of $d \times d$. The square search points are obtained by traversing the block in the upper-right, upper-left, down-left and down-right directions. Here, (x, y) is the centre point that is considered from the reference frame.

3.2. Adaptive order hexagon search

More recently, the hexagonal search has received considerable development over the other fast algorithms. The reason is that it adopts a hexagonal search pattern, which evaluates few search points, to attain faster processing. The hexagonal search enables the motion, which is caused due to abnormal scenario like, translation, zooming, pan and tilt, to be easily detected than finding the search point that is defined through the square corner. The hexagon search of centre pixels helps in finding the best matching blocks. In the hexagonal search, a total of seven search points are formed using six points from the corner and one from the centre to find the best matching block in accordance with the rate-distortion condition.

In the proposed method, the hexagonal search is modified by including the adaptive order concept and the trade-off criterion. In the adaptive order hexagon search algorithm, the centre points are moved to the search points by defining a hexagonal shape, the depth of which is described using the order. Here, the order is defined as either one or zero to handle the size of the hexagon that performs the searching procedure. The depth of the hexagon is the summation of the order of the search and a constant C, which is a parameter that is to be fixed manually. So, two sets of hexagonal points are introduced here to search and find the best matching block.

Here, the centre point is moved to the six corner points of the hexagon to discover the best matching block of the reference frame. Assume that (x, y) is the centre location of the search point in the hexagonal shape. The searching of the best matching block is done through the six corner points of the hexagon depending on the following equation. Here, the order is also an important parameter.

$$SP_1^H = (x+p, y+q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{6}$$

$$SP_2^H = (x-p, y+q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{7}$$

$$SP_3^H = (x+p, y-q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{8}$$

$$SP_4^H = (x-p, y-q); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{9}$$

$$SP_5^H = (x-2*p, y); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{10}$$

$$SP_6^H = (x+2*p, y); \qquad p = 0, \ldots, d; \quad q = 0, \ldots, d \tag{11}$$

where, SP_1^H, SP_2^H, SP_3^H and SP_4^H, SP_5^H and SP_6^H imply the six search points that are defined in the corners of the hexagon.

3.3. Tangent-weighted trade-off criterion

Designing the trade-off criterion is an important step in finding the best matching block. Once the centre pixels are moved through different shapes, the selection of the right as well as the best centre pixel is an important criterion. The selection of the block should have good correlation with the reference block and it should impart better compression performance too. A block is said to exhibit better compression performance, if that particular block requires lesser number of bits to store it and owns minimum motion. So, the formulae to decide the best block are based on two parameters, namely, the rate and the distortion. The rate parameter is designed to attain best performance by minimizing the number of storage bits. In contrast, the distortion parameter produces best performance by searching a block with minimum motion. These two criteria have led the formulation of a maximization function, which is expressed as:

$$T(b) = \eta * T_d(b) + \tau * T_r(b) \tag{12}$$

where, η and τ are the weighted constants of the parameters, T_d and T_r, respectively. The first parameter, belonging to distortion T_d (b), considers the motion-dependent parameter $M(b)$ and the weight function $W_d(b)$. This parameter $T_d(b)$ should be a maximum value, if better performance has to result and it can be stated as:

$$T_d(b) = M(b) * W_d(b) \tag{13}$$

The motion-dependent parameter $M(b)$ is computed by taking the difference ratio of the size of the block, with respect to the pixels that have static values. So, this parameter will get a maximum value, if there exists an exactly equivalent block that resembles the reference block. Otherwise, the values will be towards zero.

$$M(b) = \frac{S(b) - A(b)}{S(b)} \tag{14}$$

where, $S(b)$ refers to the number of pixels in the block and $A(b)$ indicates the number of pixels with dynamic values, as in the previous frame. The weightage for the motion-dependent parameter $M(b)$ is based on the size of the block and the tangent function, which always lies in between 0 and 1. If the size of the block is maximum, then the weightage will be one.

$$W_d(b) = \left[\frac{1}{1 + \exp[-S(b)]}\right] \tag{15}$$

The second parameter, belonging to the rate $T_r(b)$, considers the rate-dependent parameter $R(b)$ and the weight function. The rate-dependent parameter considers the bits that are required to store the block, before and after the execution of the CAVLC coding. This parameter should have a maximum value for achieving better performance.

$$T_r(b) = R(b) * W_r(b) \tag{16}$$

The rate-dependent parameter, $R(b)$, is calculated by finding the difference ratio of the bits, which are required to store the block, before and after performing the CAVLC coding. If this parameter obtains a maximum value, then the block is better in terms of the compression performance. Otherwise, the values will be towards zero.

$$R(b) = \frac{R_B(b) - R_A(b)}{R_A(b)} \tag{17}$$

where, $R_A(b)$ specifies the number of bits that are required to store the block after compression; $R_B(b)$ denotes the number of bits that are required to store the block before compression. The weightage for the rate-dependent parameter $T_r(b)$ is based on the bits, which are required to store the block after compression, and the tangent function that always lie in between 0 and 1.

$$W_r(b) = \left[\frac{1}{1 + \exp[-R_A(b)]}\right] \tag{18}$$

3.4. Design flow of the proposed AOSH Search algorithm

The design flow of the proposed AOSH algorithm is discussed in this section. Fig. 2a depicts that two sets of square points are taken for the reversal process for a fixed constant C of 1. In the first square set, five search points are visited for the order value of 0 and the next five points are visited for the same constant and the order value of 1. So, the adaptive order search will do two square searches for every searching process. Similarly, Fig. 2b shows the adaptive order hexagonal search pattern for a fixed constant C of 1. In the adaptive order hexagonal search, two sets of hexagonal search points are visited for both the order 0 and the order 1 to find the best matching block.

The flow of the proposed AOSH algorithm involves four computational steps. The algorithmic description of the proposed AOSH algorithm is given in Fig. 3.

- Step 1: Starting: In the current frame, the centre point (x, y) is located in the same location of the reference frame and a block of size BxB is formed by fixing the centre point as centre pixel.
- Step 2: Adaptive order square search: The adaptive square search is performed for the given S value, with respect to the depth and the order by traversing through the four corner points towards right, left, top and bottom directions. For all the five points that are visited in the square shape for order 0, the block is formed by fixing the current point as the centre pixel. Similarly, for the order 1, again five points will be searched to find the best matching block.
- Step 3: Adaptive order hexagonal search: Once the adaptive order square search is finished, the hexagonal search is performed to find the best matching block by traversing through the six corner points. For all the six corner points of order 0, the block is formed by fixing the current point as the centre pixel and then, the best matching block is found. Similarly, for the order 1, again six points will be searched to find the best matching block.
- Step 4: For all the pixels that are visited through the adaptive square search and hexagonal search, the trade-off criterion is found and the block with the maximum value is taken as the best matching block.

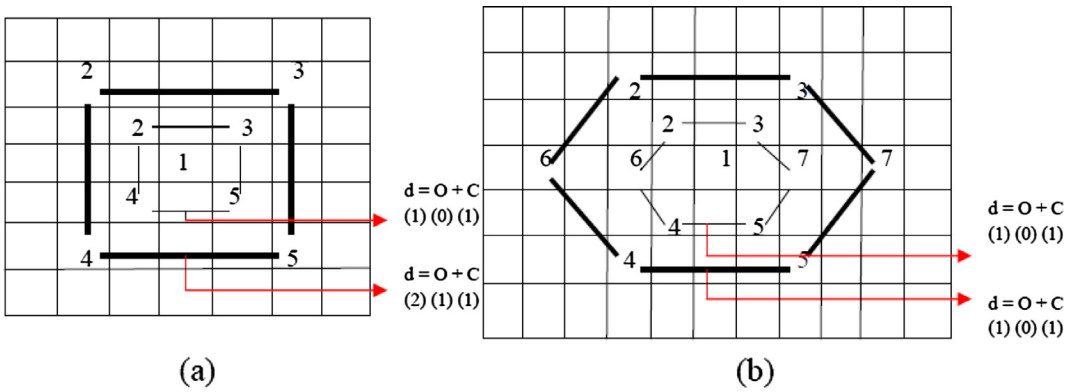

Figure 2 Search pattern (a) adaptive order square search, (b) adaptive order hexagon search.

	AOSH Algorithm
1	**Input:** Initial Point, (x, y) , Constant, C
2	**Output:** Best matching point (x_b, y_b)
3	**Procedure**
4	**Start**
5	Read initial point, (x, y)
6	Draw block for the initial point
7	Find the trade-off criterion, $T(b)$
8	For order 0 and 1
9	Find depth d
10	Get search point s $SP^S{}_1$, $SP^S{}_2$, $SP^S{}_3$ and $SP^S{}_4$ using the square search
11	Get Search points $SP^H{}_1$, $SP^H{}_2$, $SP^H{}_3$ and $SP^H{}_4$, $SP^H{}_5$ and $SP^H{}_6$ using the hexagon search
12	Find the trade-off criterion, $T(b)$ for all the search points
13	End For
14	Select the best point (x_b, y_b)
15	Return the best point (x_b, y_b)
16	**End**

Figure 3 AOSH search algorithm.

3.5. Encoding issues with the AOSH Search algorithm

The generation of encoded bits is almost similar to H.264, once the motion vectors are estimated. For a video, the important fields such as, height of the frame, width of the frame, number of frames, frame rate and scalable quantization parameters are initially encoded with 8 or 16 bits. Then, the I frame is encoded by CAVLC after doing the processes like, blocking, integer transformation and quantization. Then, every subsequent frame is encoded based on the motion vectors and the mode that results from blocking, integer transformation and quantization. Mode is used to recognize the location of the search point, as it is encoded with three bits. The first bit denotes the order of the search, the second bit denotes the direction of movement in the upward or the downward direction and the third bit denotes the direction of movement in the left or the right direction. Once the mode is inserted, the bit coding that is obtained from CAVLC is inserted. This process is continued for all the frames.

4. Results and discussion

This section describes about the experimental results of the proposed video compression approach, which relies on the adaptive order search and the tangent-weighted trade-off criterion. The performance analysis of the proposed method against the existing methods is also presented with a detailed comparative analysis using SSIM and PSNR.

4.1. Experimental set up

The proposed method is implemented using MATLAB (R2014a). The experimentation has been achieved with a system, which owns an i3 processor and a main memory of 2 GM RAM with windows 7 as operation system. The validation of the proposed method is done using three videos namely, football, garden and tennis, which have been deployed in CIPR. The video characteristics that are taken for

Table 2 Characteristics of the input videos and the algorithmic parameters.

	Characteristics of the input videos			Algorithmic parameters				
	Time (s)	Frame rate	Number of frames	Scalar value for quantization	Macro block size	Search parameter C	Weighted constant, η	Weighted constant, τ
Football	2	30	60	27	16 * 16	1	0.6	0.4
Garden	2	30	60	27	16 * 16	1	0.6	0.4
Tennis	2	30	60	27	16 * 16	1	0.6	0.4

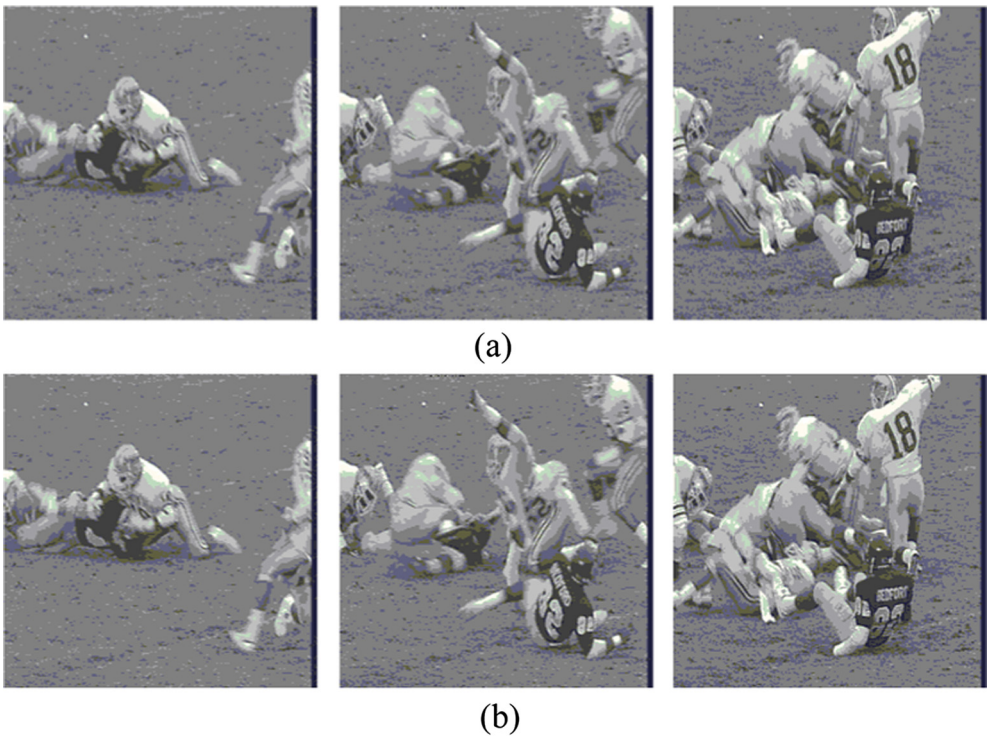

Figure 4 Football video: (a) sample frames, (b) decompressed frames.

experimentation as well as the algorithmic parameters are given in Table 2

The performance of the proposed method is analysed using SSIM (Wang et al., 2004) and PSNR. SSIM and PSNR are used to evaluate the visual quality of the video, after performing decompression. Bit rate is a parameter that is associated with the compression performance of the video. The definition of these metrics are given below.

$$SSIM(F_D,F_O)=\frac{(2*M(F_D)*M(F_O)+C_1)*(2*CV(F_D,F_O)+C_2)}{(M^2(F_D)+M^2(F_O)+C_1)*(V(F_D)+V(F_O)+C_2)} \tag{19}$$

where, $M(F_D)$ points to the mean of the decompressed frame F_D, $M(F_O)$ represents the mean of the original frame F_O, $V(F_D)$ stands for the variance of the decompressed frame F_D, $V(F_O)$ indicates the variance of the original frame F_O, $CV(F_D, F_O)$ denotes the co-variance of the frames F_D and F_O and finally, C_1 and C_2 represent the constants.

$$PSNR=10\log_{10}\frac{E^2_{\max}\times I_w\times I_h}{\sum\sum\left(I_{xy}-I^*_{xy}\right)^2} \tag{20}$$

where, I_w and I_h orderly denote the width and the height of the frame, I_{xy} specifies the original pixel value at coordinate (x, y), I^*_{xy} refers to the decompressed pixel value at coordinate (x, y) and $E^2_{\max}$ stands for the largest energy of the pixels.

4.2. Experimental results

This section shows the experimental results of the proposed compression system. Figs. 4a, 5a and 6a portray the sample frames, which are taken from three different videos such as, football, garden and tennis. Figs. 4b, 5b and 6b reveal the decompressed frame of the three different videos. The original video as well as the decompressed video has an increased visualization similarity, without much visual degradation for the human visual system.

4.3. Evaluation of visual quality using SSIM

The performance of the proposed compression system in terms of visual quality is evaluated using SSIM and a comparative analysis with the existing work of Muhit et al., (2010) and H.264 is also performed. Fig. 7 shows the SSIM graph for the football video. From the figure, it has been ensured that the proposed algorithm obtains a SSIM value of 0.92 for the lower bit rate as well as the higher bit rate. In contrast, the

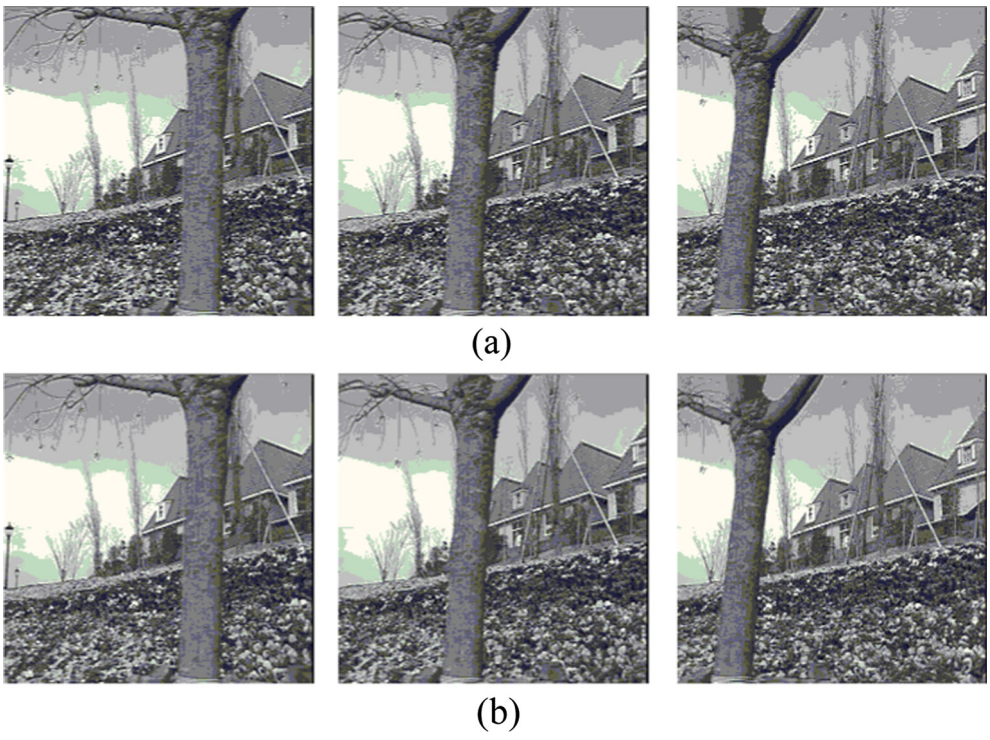

Figure 5 Garden video: (a) sample frames, (b) decompressed frames.

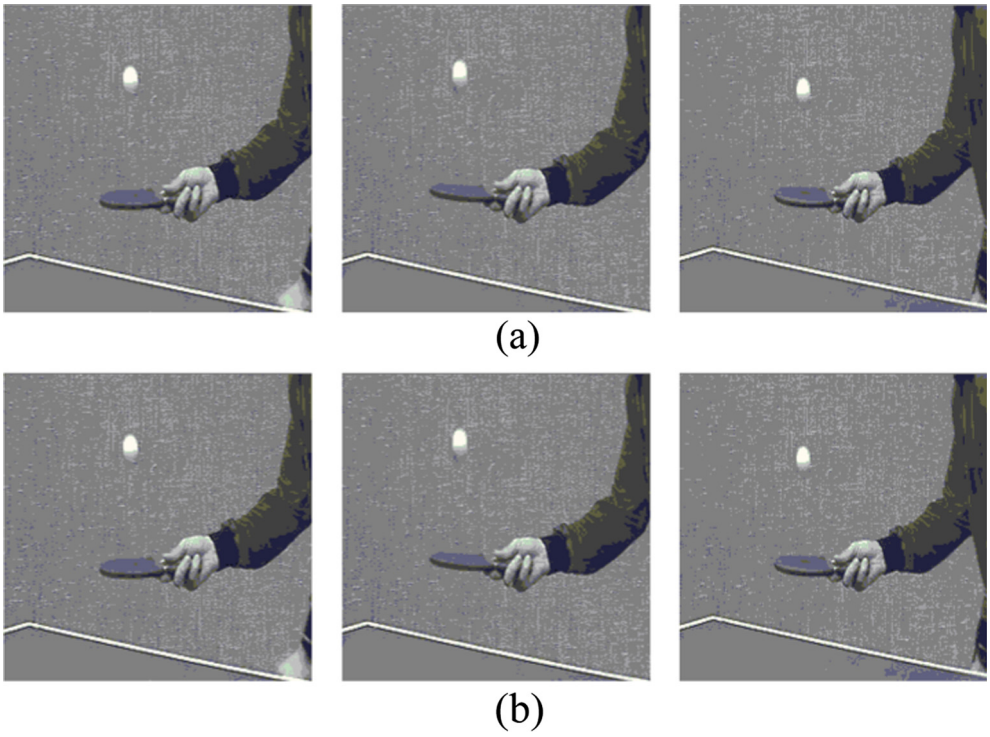

Figure 6 Tennis video: (a) sample frames, (b) decompressed frames.

elastic model has achieved a value of 0.64 for the lower bit rate. Fig. 8 shows the SSIM graph for the garden video. The figure illustrates that the proposed algorithm has obtained a value of 0.97 for the higher bit rate. H.264 and the elastic model have obtained a value of just 0.74, which again ensures that the proposed algorithm exhibits supreme performance in terms of visual quality. Fig. 9 shows the SSIM graph for the tennis video. The proposed system has attained a value of 0.93 for

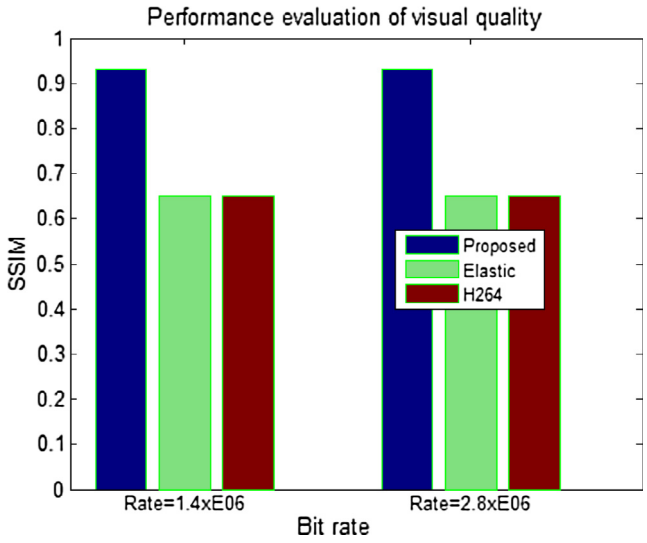

Figure 7 SSIM graph for football video.

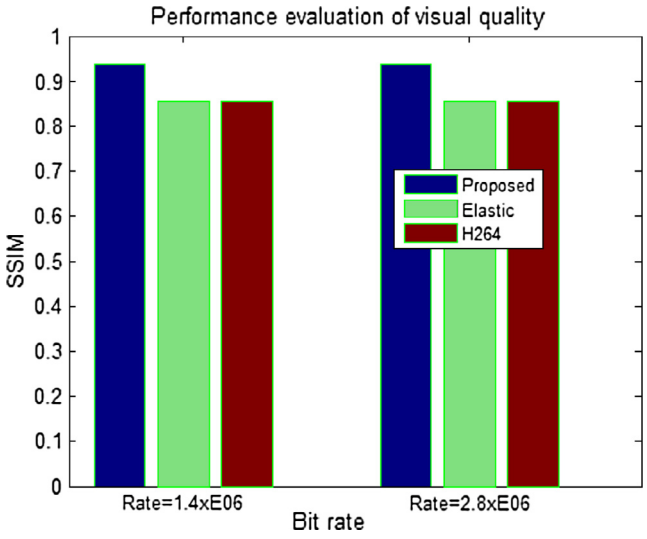

Figure 9 SSIM graph for tennis video.

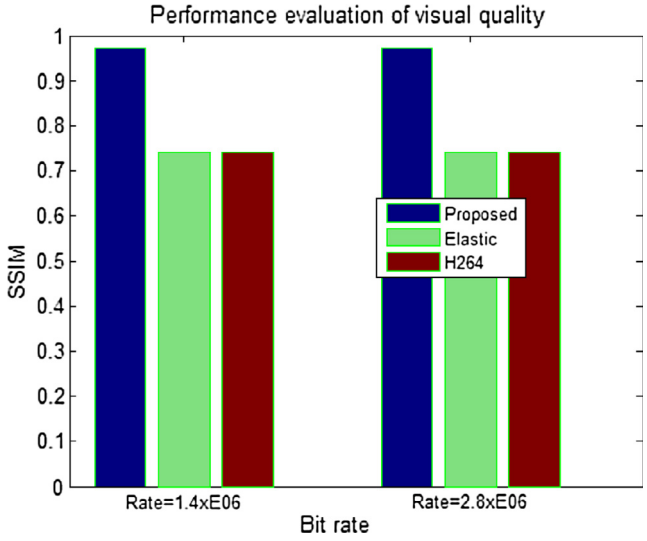

Figure 8 SSIM graph for garden video.

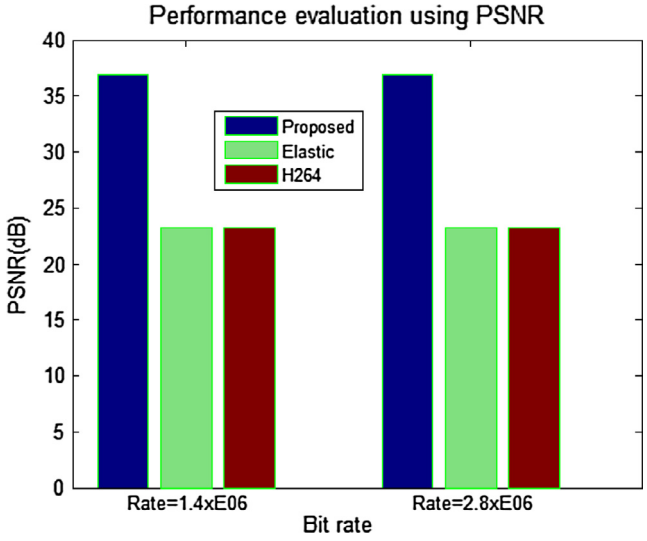

Figure 10 PSNRgraph for football video.

the tennis video, but the existing algorithms have reached only 0.85. All these analysis results have ascertained the dominant performance of the proposed algorithm in terms of visual quality.

4.4. Evaluating the performance using PSNR

This section presents the performance analysis of the proposed algorithm using PSNR, with respect to the elastic model and H.264. Fig. 10 shows the PSNR graph for the football video. Here, the proposed algorithm has orderly achieved 36.89 dB and 36.88 dB for the lower and the higher bit rates. But, the elastic model and H.264 has attained a value of 23.19 dB and H.264, respectively. In the garden videos, the proposed algorithm has obtained the PSNR value of 36.86 dB for the lower bit rate and 36.96 dB for the higher bit rate as in Fig. 11. The elastic model has reached 20.15 dB and 20.15 dB for the lower bit rate and the higher bit rate, respectively. The H.264 model has obtained 20.15 dB and 20.15 dB

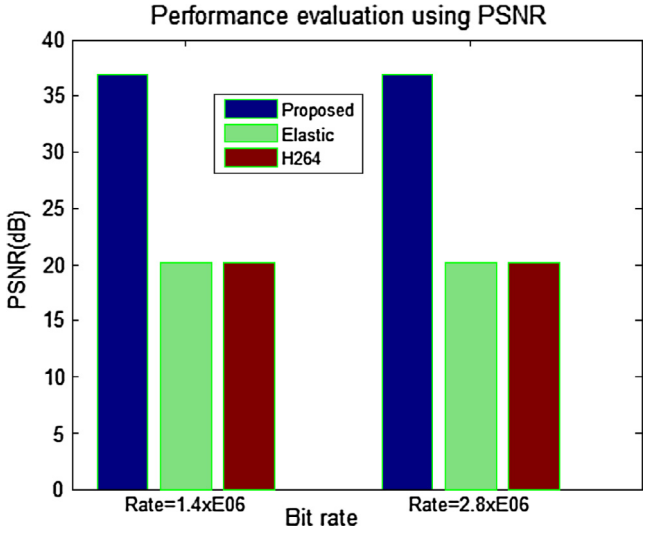

Figure 11 PSNRgraph for garden video.

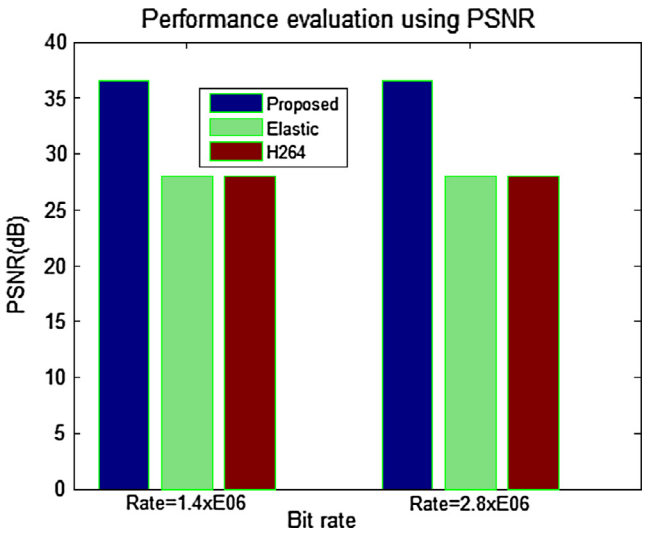

Figure 12 PSNRgraph for tennis video.

for the lower bit rate and the higher bit rate. This ensures that the proposed model exhibits better performance than the existing algorithms, in terms of the visual performance. Fig. 12 shows the PSNR graph for the tennis video. In this case, the proposed algorithm, the elastic model and H. 264 model have obtained the PSNR value of 36.52 dB, 27.99 dB and 27.99 dB, respectively, for lower bit rate. From the analysis, it is evident that the proposed algorithm is better than the compared existing algorithms for video compression.

5. Conclusion

This paper has presented an adaptive order search and tangent-weighted trade-off for achieving motion estimation in H.264. Here, the AOSH algorithm has been developed by combining the square as well as the hexagon search. This has improved the searching capability of the motion estimation processes with less computation complexity. In addition, the tangent- weighted trade-off has been deigned to evaluate the search points using two parameters called, the rate and the distortion. These two enhancements have been applied to the block estimation process of H.264 to improve the visual quality as well as the compression performance. The experimentation has been performed using three videos, namely, football, garden and tennis. Then, the performance of the proposed method has been compared with that of H.264 and the Elastic model using two quantitative parameters, namely, SSIM and PSNR. The proposed method excels the existing methods for these two quantitative parameters and, in turn, has ensured supreme visual quality and compression performance. In future, the proposed algorithm can be further improved by including multiple constraints, other than the rate and the distortion.

References

Advanced Video Coding for Generic Audiovisual Services, ITU-T Rec.H.264 and ISO/IEC 14496-10 AVC, 2003.

Bosch, Marc, Zhu, Fengqing, Delp, Edward J., 2011. Segmentation-based video compression using texture and motion models. IEEE J. Sel. Top. Signal Process. 5 (7), 1366–1377.

Chen, Xiaolin, Canagarajah, Nishan, Nunez-Yanez, Jose L., Vitulli, Raffaele, 2012. Lossless video compression based on backward adaptive pixel-based fast motion estimation. Signal Process. Image Commun. 27, 961–972.

CIPR sequences from <http://www.cipr.rpi.edu/resource/sequences/sif.html>.

Duanmu, Chunjiang, Zhang, Yu, 2012. A new fast block motion algorithm based on octagon and triangle search patterns for H.264/AVC. JDCTA 6 (10), 369–377.

Dufaux, F., Moscheni, F., 1995. Motion estimation techniques for digital TV: a review and a new contribution. Proc. IEEE 83 (6), 858–876.

Fabrizio, Jonathan, Dubuisson, Séverine, Béréziat, Dominique, 2012. Motion compensation based on tangent distance prediction for video compression. Signal Process. Image Commun. 27 (2), 153–171.

Huang, S., Hsieh, B., Chien, S., Ma, S., Chen, L., 2006. Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC. IEEE Trans. Circuits Syst. Video Technol. 16 (4), 507–522.

ITU-T and ISO/IEC JTC 1, 2010. Advanced video coding for generic audiovisual services. In: ITU-T recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 AVC).

Liu, Pengyu, Gao, Yuan, Jia, Kebin, 2014. An adaptive motion estimation scheme for video coding. Sci. World J. Article ID 381056, 14p.

Ma, S., Kuo, C.-C.J., 2007. High-definition video coding with supermacroblocks. In: Proc. SPIE Vis. Commun. Image Process., vol. 6508, pp. 650816-1–650816-12.

Mathew, R., Taubman, D., 2006. Hierarchical and polynomial motion modeling with quad-tree leaf merging. In: Proc. Int. Conf. Image Process., pp. 1881–1884.

Muhit, Abdullah A., Pickering, Mark R., Frater, Michael R., Arnold, John F., 2010. Video coding using elastic motion model and larger blocks. IEEE Trans. Circuits Syst. Video Technol. 20 (5), 661–672.

Muhit, Abdullah A., Pickering, Mark R., Frater, Michael R., Arnold, John F., 2012. Video coding using fast geometry-adaptive partitioning and an elastic motion model. J. Vis. Commun. Image R. 23, 31–41.

Pan, Z., Kwong, S., 2011. A fast inter-mode decision scheme based on luminance difference for H.264/AVC. In: Proc ICSSE'11, pp. 260–263.

Pan, Z., Kwong, S., Xu, L., Zhang, Y., Zhao, T., 2012. Predictive and distribution-oriented fast motion estimation for H.264/AVC. J. Real-Time Image Proc.

Pan, Zhaoqing, Kwong, Sam, 2013. A direction-based unsymmetrical-cross multi-hexagon-grid search algorithm for H.264/AVC motion estimation. J. Signal Process Syst. 73, 59–72.

Sullivan, G.J., Wiegand, T., 1998. Rate-distortion optimization for video compression. IEEE Signal Process. Mag. 15 (6), 74–90.

Video Codec for Audiovisual Services at p ×64 kbits, ITU-T Rec. H.261, 1993.

Video Coding for Low Bitrate Communication, Version 1, ITU-T Rec. H.263, 1995.

Wang, H., Kwong, S., 2008. Rate-distortion optimization of rate control for H.264 with adaptive initial quantization parameter determination. IEEE Trans. Circuits Syst. Video Technol. 18 (1), 140–144.

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13 (4), 600–612.

Wiegand, T., Sullivan, G.J., Bjontegard, G., Luthra, A., 2003. Overview of the H.264/AVC video coding standard. IEEE Trans. Circ. Syst. Video Technol. 13 (7), 560–576.

Zhao, T., Wang, H., Kwong, S., Kuo, C.-C.J., 2010. Fast mode decision based on mode adaptation. IEEE Trans. Circuits Syst. Video Technol. 20 (5), 697–705.