A Model-based Adaptive Motion Estimation Scheme Using Renyi's Entropy for Wireless Video

Ganesan Ramachandran [a] Vignesh Krishnan [a] Dapeng Wu [a,1]
Zhihai He [b]

[a] *Department of Electrical & Computer Engineering, University of Florida, Gainesville, FL 32611-6130, USA.*

[b] *Department of Electrical & Computer Engineering, University of Missouri, Columbia, MO 65203*

Abstract

Efficient motion estimation is an important problem because it determines the compression efficiency and complexity of a video encoder. Motion estimation can be formulated as an optimization problem; most motion estimation algorithms use mean squared error (MSE), sum of absolute differences (SAD) or maximum a posteriori probability (MAP) as the optimization criterion and apply search-based techniques (e.g., exhaustive search or three-step search) to find the optimum motion vector. However, most of these algorithms do not effectively utilize the knowledge gained in the search process for future search efforts and hence are computationally inefficient. This paper addresses this inefficiency problem by introducing an adaptive motion estimation scheme that substantially reduces computational complexity while yet providing comparable compression efficiency, as compared to existing fast-search methods.

Our approach is motivated by the recent developments in using Renyi's entropy as the optimization criterion for system modeling [1]. This scheme is particularly suited for wireless video sensor networks, video conferencing systems and live streaming videos which have stringent computational requirements. Our results show that our scheme reduces the computational complexity by a factor of 9 to 21, compared to the existing fast algorithms.

Key words: Adaptive motion estimation, Renyi's entropy, wireless video, model-based motion estimation, very low complexity.

1 Introduction

Video communication has seen a tremendous surge in the past decade. There is a lot of interest in transmission of digital video over the Internet. Transmission of digital video requires extremely high data rates. For example, a raw video with a resolution of 176 x 144 pixels/frame and a frame rate of 10 frames/s has a data rate of 2 Mb/s. Consequently, efficient video compression must be in place. There have been significant advances in digital multimedia compression algorithms, which make it possible to deliver high-quality video at relatively low bit rates.

Hybrid video coding [2] is the most widely used video coding scheme. It uses a combination of temporal prediction (inter-coding) and transform (intra) coding to achieve good coding efficiency. Inter-coding exploits the temporal redundancy between frames using motion estimation and compensation. The motion estimation requires the specification of the underlying model (camera, illumination, object, scene, motion), an estimation criterion (displaced frame difference, optical flow equation, Bayesian criterion), optimization criterion (Lp norm, entropy, correlation) and a search strategy (stochastic or deterministic) [2,3].

For wireless video applications, a key requirement is low computational complexity. Since in most cases motion estimation constitutes roughly 70% of the computational load on the video encoder [4], it is a major concern for designing wireless video applications. Thus, there is a need for a fast, simple and efficient motion estimation algorithm.

In this paper, we propose an adaptive motion estimation approach to model motion vectors for inter-coding. The objective of our adaptive motion estimation scheme is to achieve good quality video with very low computational complexity. Our method attempts to model the motion vectors using adaptive filtering techniques. The main contributions of this paper are (1) an optimization criterion for motion estimation, which uses an information theoretic framework rather than traditional variance-based framework, and (2) a very low complexity search algorithm ideally suited for wireless video applications.

The remainder of the paper is organized as follows. Section 2 discusses the related works on motion estimation and identifies their limitations. To address

Email addresses: `rganesan@ufl.edu` (Ganesan Ramachandran), `vigneshk@ufl.edu` (Vignesh Krishnan), `wu@ece.ufl.edu` (Dapeng Wu), `HeZhi@missouri.edu` (Zhihai He).

URL: `http://www.wu.ece.ufl.edu` (Dapeng Wu).

[1] Corresponding author Tel. (352) 392-4954, Fax (352) 392-0044, Email: wu@ece.ufl.edu

these limitations, we propose a model-based motion estimation approach. To facilitate the description of our algorithm, Section 3 introduces the types of existing adaptive systems and tries to make a choice of the algorithm that realizes our model-based motion estimation approach. In Section 4, we present our model-based adaptive motion estimation algorithm using Renyi's entropy. The unique property of our algorithm is its low complexity. Therefore, in Section 5, we analyze the complexity of the encoder and decoder using our scheme and compare it with existing schemes. In Section 6, we present the simulation results of our algorithm for varying frame rates and different test videos, and compare it with the existing schemes in terms of peak signal-to-noise ratio (PSNR). Section 7 concludes the paper and points out future research directions.

2 Related Work

We first discuss the exhaustive search method in Section 2.1 and then review existing fast algorithms in Section 2.2.

2.1 Exhaustive Search Method

In this scheme, each video frame is divided into blocks. The block is predicted from a previously coded reference frame using block-based motion estimation. The motion vectors specify the displacement between the current block and the best matching block. The predicted block is obtained from the previous frame based on the estimated motion vector (MV) using motion compensation. Then the prediction error block is coded by applying the DCT, quantizing the DCT coefficients and converting them into binary codewords using entropy coding techniques. The quantization of the DCT coefficients is controlled by the quantization parameter which scales a predefined quantization table as defined in JPEG standard [5]. The block diagram of a typical block based hybrid video encoder is shown in Fig. 1.

Block matching motion estimation algorithms work by dividing a video frame into segments of smaller size called blocks, each of which consisted, in general, of 8 x 8 pixels. The algorithm assumes that all the pixels in the block undergo the same translation. The same motion vector is assigned to all the pixels in the block. The motion vector is estimated by searching for the best matching block within a search window, centered on the corresponding block in the reference frame.

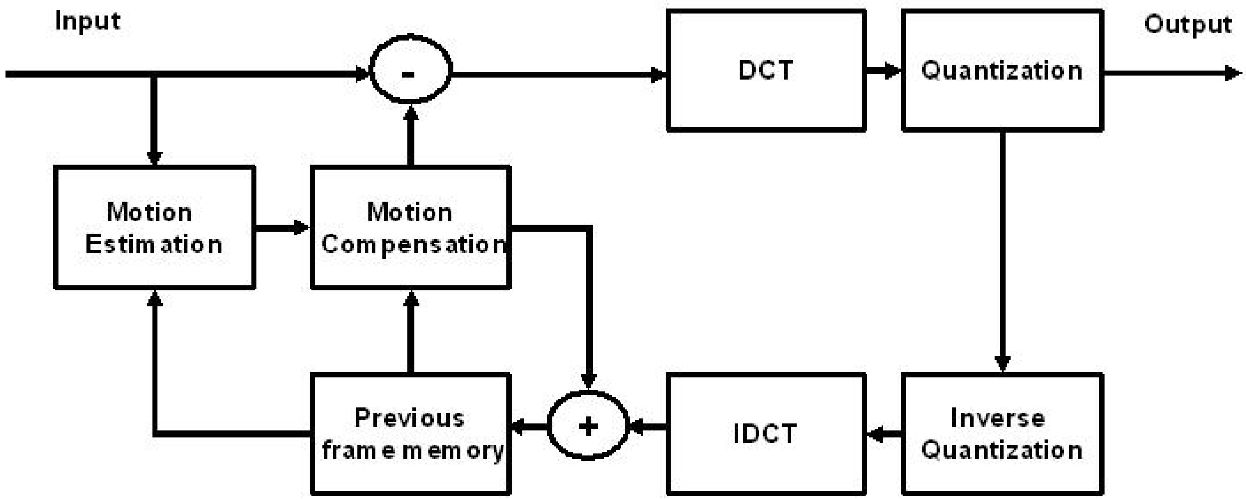

Fig. 1. Block diagram of the encoder.

The process is formulated as follows:

$$E({d_n}^{(x)}, {d_n}^{(y)}, \forall m \in M) = \sum_{\forall m \in M} \sum_{x,y \in B_m} BDM\left(f(x,y,n+1) - \widetilde{f}(x + {d_n}^{(x)}, y + {d_n}^{(y)}, n)\right) \quad (1)$$

$$[\hat{d}_n^{(x)}, \hat{d}_n^{(y)}] = arg \min_{{d_n}^{(x)}, {d_n}^{(y)}} E({d_n}^{(x)}, {d_n}^{(y)}, \forall m \in M) \quad (2)$$

$$\hat{f}(x,y,n+1) = \widetilde{f}(x + \hat{d}_n^{(x)}, y + \hat{d}_n^{(y)}, n) \quad (3)$$

$$e(x,y,n+1) = f(x,y,n+1) - \hat{f}(x,y,n+1) \quad (4)$$

$$e'(x,y,n+1) = Q[e(x,y,n+1)] \quad (5)$$

$$\widetilde{f}(x,y,n+1) = \hat{f}(x,y,n+1) + e'(x,y,n+1) \quad (6)$$

where is f the original video frame, $\widetilde{f}$ if the motion compensated frame, $\hat{f}$ is the motion estimated frame, e is the estimation error and e' its quantized value, $Q[\cdot]$ denotes the quantization operator, n is the time reference, ${d_n}^{(x)}$ is the motion vector in the x-direction, ${d_n}^{(y)}$ is the motion vector in the y-direction, B_m is the m^{th} block, M is the number of blocks and BDM is some block distortion measure. It can be Sum of Squared Differences (SSD), Sum of Absolute Differences (SAD), cross-correlation function, to mention a few.

When equation (1) is evaluated for all possible pixel locations, the algorithm is called the exhaustive block matching algorithm (EBMA). This is the optimal solution for a given block size and search window. However, the computational complexity of EBMA is very high.

2.2 Fast Search Algorithms

In most practical cases, an exhaustive search may not be required as the motion is not completely random. In the literature, there are several classes of algorithms that perform motion estimation with reduced computational complexity, as compared to their exhaustive counterparts. We list four major classes of fast motion estimation algorithms as below.

The first class of fast algorithms works by reducing the search to a subset of blocks which are more likely to be the desired block that provides the minimum BDM. The main assumption is that the error surface is uni-modal, which means that there is only one local minimum, which is also the global minimum. Some of the algorithms that belong to this class are Two-dimensional logarithmic (TDL) search [6], block-based gradient descent search [7], three-step search [8], a new three-step search (TSS) [9], the four-step (4SS) search [10], to mention a few. But the error surface is usually multi-modal, i.e., it contains many local minima due to frame skipping, motion and video content. The global minimum of the error surface can change due to these factors.

The second type of fast algorithms [11] is based on the assumption that the block-motion fields are usually smooth. This implies that there is a high probability that the neighboring blocks will have almost the same motion vectors. A sub-sampled block motion field is obtained by applying motion estimation to a subset of blocks in the frame. The motion field is then interpolated to get the complete motion vector information for the frame. This type of algorithms reduces the computational complexity but suffer from poor prediction quality.

The third category of fast algorithms utilizes multi-resolution search techniques, which can combat two major limitations of aforementioned fast search algorithms. First, the error surface is usually multi-modal and hence they may not reach the global minimum; and second, the computational complexity involved in the calculation of the BDM function is very high. These can be overcome by the multi-resolution method, which searches the solution space of an optimization problem at successively finer resolutions. The search proceeds by first obtaining a rough estimate of the solution at a lower resolution which is easier to evaluate. This solution is fed to the next stage which is performed at a higher resolution and so on till the highest resolution is reached. The search window is also reduced at each step. Examples of this scheme are [12] and [13]. The limitation of this type of schemes is that it requires increased storage because we need to keep images at several resolutions. This method can yield inaccurate results for videos with small objects because it performs the search at a lower resolution.

The fourth kind of fast algorithms is based on fast full search techniques. Instead of reducing the computational complexity of block matching at the expense of prediction quality by restricting the search space, fast full search algorithms are intended to reduce computational complexity without sacrificing prediction quality. They usually achieve this by evaluating the BDM function and comparing it with the lowest BDM value calculated so far. If the BDM value of the candidate exceeds the known lower bound, it is discarded and the search proceeds to the next candidate. Examples of the algorithms like this are Partial Distortion Elimination [14], Successive Elimination Algorithm [15], and Winner-Update Strategy [16]. However, the problem with these algorithms is the unpredictable amount of computation. In noisy video sequences, or even sequences with a large amount of motion, these algorithms reject only small part of the candidate motion vectors and do not provide much saving in computation.

We notice that the aforementioned four types of algorithms do not effectively utilize the knowledge gained in calculating the motion vectors from one frame to the next. These algorithms do not track the motion estimation along the frames. For each search, they usually reset their memory and start from the same initial conditions (some of them may use a median filter to obtain initial values of motion vectors). We propose a novel model-based scheme which takes advantage of the information gained from past frames to estimate the motion vectors of future frames. Though similar attempts have been made in the past, they usually decoupled the motion estimation from the prediction of the motion vectors [17]. The goal is to develop an adaptive system that models the motion field rather than code the motion field directly. The advantage of our scheme is that the motion estimation is very fast. Since the motion model can be replicated at the decoder given knowledge about the initial conditions there is no need to transmit motion vectors. This provides savings in bandwidth. The extra bits can be used for the transmission of residual errors and thus providing better reconstruction.

Thus far we have identified the advantages of a model-based adaptive scheme in terms of fast motion estimation and bandwidth savings. In the next section, we make a choice of the adaptive algorithm that we use for the motion estimation and also introduce our choice of optimization criterion for adaptation and justify its selection.

3 Choice of the adaptive system

Adaptive filters have been successfully used in various research areas including signal processing, telecommunication, system identification, and automated control. The efficiency of the adaptive filters is rooted in the optimal estimation theory, which allows the filter to automatically adapt to the minimum mean squared error (MMSE) solution efficiently. Least mean square (LMS) algorithm and recursive least square (RLS) algorithm are the two best known techniques to approximate the optimal Wiener filter solution. Both LMS and RLS recursively compute the optimal filter's weights, resulting in simple implementation, fast convergence rate, and good performance.

Mean square error (MSE) has been the most widely used criterion for the training of most adaptive systems. The two main reasons for this choice were: the simplicity in computation, and the Gaussian probability density assumption. It has been shown in [1] that MSE may not be the best criterion because most real-life problems are governed by nonlinear equations and most random phenomena are far from being normally distributed. Therefore, the training of adaptive systems requires a criterion other than MSE which takes into account not only second-order statistics like MSE but also the higher-order statistical behavior of the systems.

Shannon's entropy of a given probability density function (pdf) is a scalar quantity that provides a measure of the average information content of any distribution. Information is a function of the pdf itself and hence the entropy is related to the pdf rather than any particular statistics of it. Therefore it is a more useful criterion for training adaptive systems than MSE.

Given the current video frame and the next frame, the adaptive system can predict the motion vectors by minimizing the entropy of the estimation error similar to the block matching methods which tries to find the best matching block by minimizing the SSD or SAD. The interpretation of this is as follows. When entropy of the error is minimized, the expected information contained in the estimation error is minimized; hence the adaptive system is trained optimally in the sense that the mutual information between the video input and the model output is maximized. However, there are no analytical methods to use Shannon's entropy in adaptation. Hence, we use a nonparametric entropy estimator based on Renyi's entropy [18], which can be applied directly to data samples collected from experiments and without imposing any a priori assumptions about the pdf of the data. Thus the method can manipulate information as straightforwardly as the mean square error (MSE) criterion. Straightforward methods that use the Quadratic Renyi's entropy in a way similar to LMS method have been developed in [19]. We follow this approach for estimation of motion in video frames.

Fig. 2. A snapshot of the original image of the video sequence "Suzie".

We show one snapshot of the motion estimation error images of the test video sequence "Suzie" for our adaptive system with MSE and Entropy as criteria for adaptation. Fig. 2 shows one original frame of the test video sequence "Suzie". From Fig. 4, it is clear that although the motion estimation error for the MSE method has smaller variance compared to the Entropy method in Fig. 3, it contains more information about the original image. The intuition is that in our scheme we try to minimize the entropy of the motion estimation error; in other words, we try to minimize the correlation between the error image and the original image. By minimizing the entropy, we minimize the useful information contained in the error image, thereby extracting the maximum possible information from the original image. This process is like whitening the error image, i.e., making the error image statistically independent of the original image.

4 Entropy based Motion Estimation scheme

The organization of this section is as below. We first formulate the motion estimation as an adaptive prediction problem in Section 4.1. Then, we introduce the general notion of Renyi's Entropy in Section 4.2 and finally develop it as a criterion for the adaptive motion estimation system for video coding in Section 4.3.

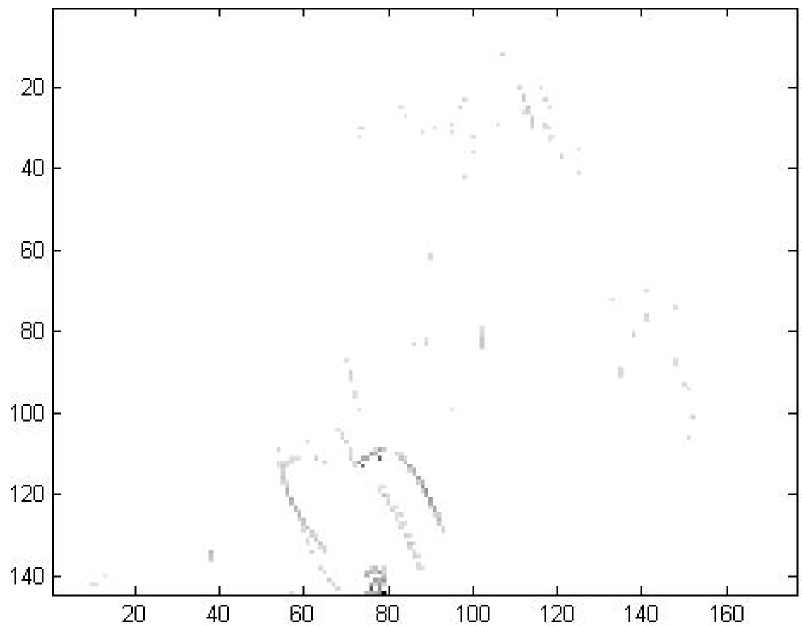

Fig. 3. A snapshot of error image of the video sequence "Suzie" using Renyi's Entropy

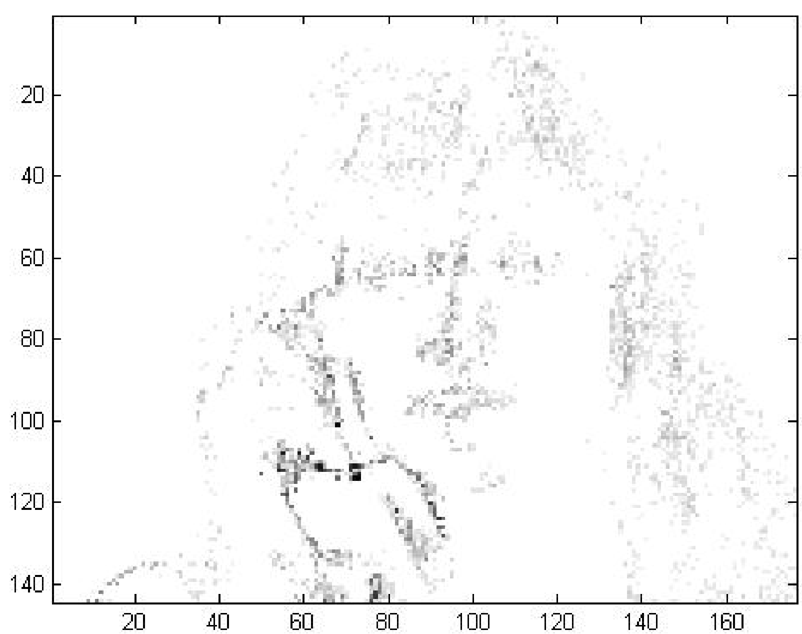

Fig. 4. A snapshot of error image of the video sequence "Suzie" using MSE

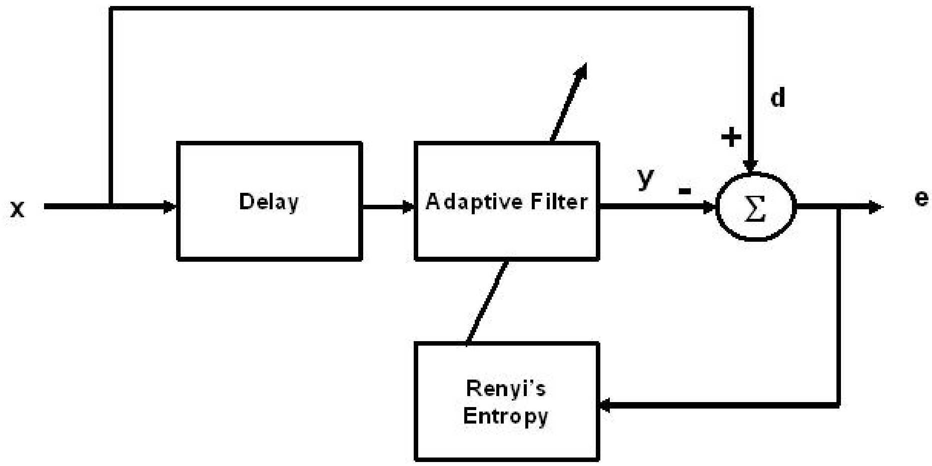

Fig. 5. Adaptive predictor

4.1 Adaptive Motion Estimation Approach

Consider the block diagram of a general adaptive prediction system in Fig. 5. The filter consists of a linear structure with an impulse response denoted by the weight vector, w. The filter input is the past values of the input, x. The present value of the input serves as the desired response, d. At some discrete time, n, the filter produces an output, y. This output is the best estimate of the current value of x given its past values, in the sense that the entropy of the estimation error, e, defined as the difference between the filter output, y and the desired response, d is minimized.

The equations representing the above problem are given as follows:

$$y = w^T x \tag{7}$$

$$e = d - y \tag{8}$$

$$\nabla_n = \frac{\partial J}{\partial w_n} \tag{9}$$

$$w_{n+1} = w_n + \mu \nabla_n \tag{10}$$

where ∇_n is the gradient of the optimization criterion, J and μ is the step size used for updating the weight equation. This is similar to the popular LMS [20] algorithm.

Translating this problem for motion estimation we define our new goal:

The objective is to design an adaptive system whose output is the current frame, f given the motion compensated frame, $\widetilde{f}$ as input, such that the entropy of the motion estimation error, e defined as the difference between the intensity of the current frame, f, and the motion compensated frame, $\widetilde{f}$, is minimized.

The function of the adaptive filter is to provide the best estimate or prediction of the current frame given knowledge of the previous motion compensated frame. Hence, the current frame serves as a desired response to the adaptive system. The previous frame is used as the input to the above adaptive system. The equations for the motion estimation problem can be formulated as follows:

$$\hat{f}(x, y, n+1) = \widetilde{f}(x + \hat{d}_n^{(x)}, y + \hat{d}_n^{(y)}, n) \tag{11}$$

$$e(x, y, n+1) = f(x, y, n+1) - \hat{f}(x, y, n+1) \tag{12}$$

$$e'(x, y, n+1) = Q[e(x, y, n+1)] \tag{13}$$

$$\widetilde{f}(x, y, n+1) = \hat{f}(x, y, n+1) + e'(x, y, n+1) \tag{14}$$

$$\hat{d}_{n+1}^{(x)} = \hat{d}_n^{(x)} + \mu \nabla_n{}^x \tag{15}$$

$$\hat{d}_{n+1}^{(y)} = \hat{d}_n^{(y)} + \mu \nabla_n{}^y \tag{16}$$

where $f$is the original video frame, $\widetilde{f}$ if the motion compensated frame, $\hat{f}$ is the motion estimated frame, e is the estimation error and e' its quantized value, Q[$\cdot$] denotes the quantization operator, n is the time reference, $\hat{d}_n^{(x)}$ is the motion vector in the x direction, $\hat{d}_n^{(y)}$ is the motion vector in the y direction, μ is the adaptation parameter, $\nabla_n{}^x$ and $\nabla_n{}^y$ the stochastic gradient estimate in the x and y directions respectively.

Our hypothesis is that for the same initial conditions, both encoder and decoder can estimate the motion vectors in real-time thereby avoiding the need for transmitting them. But note that in usual LMS, we need to update the motion vectors using the difference between the true motion vectors and the estimated ones. Since we don't have the true motion vectors, we use the motion estimation error itself to update the motion vectors. Also we require lossless transmission of the error signal e, for the best estimation. But in reality due to bandwidth restrictions the error signal e is quantized to . This results in sub-optimal estimates of the motion vectors. Also packet losses can lead to error accumulation. This can be minimized by introducing intra frames periodically. In this paper we address the issue of motion estimation only. Issues regarding robust transmission will be omitted here and set aside for future work.

From equations (11) to (16), we know everything required to perform motion estimation except how to calculate the gradients namely $\nabla_n{}^x$ and $\nabla_n{}^y$. For this we first need to introduce the Renyi's entropy and the information potential criterion [21] for batch adaptation. Next, we define the stochastic information gradient algorithm from [19] and show how it can be used as a criterion for the adaptive motion estimation problem.

4.2 Renyi's Entropy

Information theory is a mathematical framework for defining information content of signals. Renyi's entropy is an alternative entropy measure similar to the well known Shannon's entropy. Without knowledge of the underlying pdf, Shannon's entropy is difficult is estimate using simple algorithms as compared to Renyi's entropy. Renyi's entropy has been successfully applied to many problems in physics, signal processing and pattern recognition.

Renyi's entropy for a random variable e is defined in [18] as

$$\mathrm{H}_\alpha(e) = \frac{1}{1-\alpha} \log \int f^\alpha(e) de, \alpha \neq 1 \tag{17}$$

where $f^\alpha(e)$ is the pdf of the variable e and $\alpha > 0$ is the order of entropy. As $\alpha \to 1$, Renyi's entropy becomes Shannon's entropy.

The argument of the log is defined as the (order-α) information potential [21]. The information potential can be written as an expected value as follows:

$$\mathrm{V}_\alpha(e) = \int f^\alpha(e) de = E[f^{\alpha-1}(e)] \approx \frac{1}{N} \sum_{i=1}^{N} f^{\alpha-1}(e_i) \tag{18}$$

The pdf in equation (18) can be obtained by Parzen window estimation [22], resulting in a nonparametric estimator of information potential

$$\mathrm{V}_\alpha(e) = \frac{1}{N^\alpha} \sum_j \left(\sum_i \kappa_\sigma(e_j - e_i) \right)^{\alpha-1} \tag{19}$$

where κ_σ is the kernel function in Parzen windowing and σ, the width of the window. Writing the σ-wide kernel in terms of a unit width kernel, κ we get

$$\kappa_\sigma(x) = \frac{1}{\sigma} \kappa\left(\frac{x}{\sigma}\right) \tag{20}$$

For $\alpha = 2$ in equation (19) reduces it to

$$\mathrm{V}_2(e) = \frac{1}{N^2} \sum_j \sum_i \kappa_\sigma(e_j - e_i) \tag{21}$$

Substituting $\alpha = 2$ in equation (17) becomes

$$\mathrm{H}_2(e) = -\log \int f^2(e) de \tag{22}$$

This is known as Quadratic Renyi's Entropy (QRE) which we will be using subsequently for further considerations.

We have represented Renyi's Entropy in terms of the information potential. Notice that the log operator in equation (19) is monotonic. This will help us in the next section where instead of minimizing Renyi's entropy to obtain certain system parameters, we can maximize the information potential for values of $\alpha > 1$. Therefore, the information potential can replace the entropy criterion in adaptation with significant computational savings.

4.3 Stochastic gradient estimator

Consider that the adaptive system shown in Figure 4. We follow the derivation in [19] for developing an online adaptation algorithm for the case when $\alpha = 2$ (Quadratic Renyi's Entropy). The sample error of the system can be represented by $e_n = d_n$ - y_n. Then the instantaneous gradient of the information potential estimator at time instant n with respect to the weight vector w is given by

$$\left(\frac{\partial V_2}{\partial w}\right)_n = -\frac{1}{2} \kappa'_\sigma(e_{n,n-1}) \frac{\partial y_{n,n-1}}{\partial w} \tag{23}$$

The information potential can be maximized by using this instantaneous gradient estimator as the update in the steepest ascent algorithm.

$$w_{n+1} = w_n + \eta \left(\frac{\partial V_2}{\partial w}\right)_n \tag{24}$$

The instantaneous gradient given in equation (23) requires a single evaluation of the derivative of the kernel function at the point $e_{n,n-1} = e_n - e_{n-1}$. If a

Gaussian kernel function is chosen, its derivative is simply a product of its argument and the Gaussian evaluated at the same point again.

$$\kappa_\sigma(e_i) = G(e, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{{e_i}^2}{2\sigma^2}\right) \tag{25}$$

where σ is the variance of the error signal.

In the case of FIR filters, equation (25) can be further simplified by replacing the gradient of the output with respect to the weight vector by simply the input vector.

$$\frac{\partial y_{n,n-1}}{\partial w} = \frac{\partial y_n}{\partial w} - \frac{\partial y_{n-1}}{\partial w} = x_n - x_{n-1} \tag{26}$$

Thus we have all the information necessary to evaluate the equation (24), the weight update equation needed for successful adaptation.

For adaptive motion estimation the value of the kernel function and the stochastic instantaneous gradient for updating the motion vectors in equations (15) and (16) can be obtained from the equations (23) to (26). The equations are given as follows:

$$e_D(x, y, n+1) = e'(x, y, n+1) - e'(x, y, n) \tag{27}$$

$$e_R(x, y, n+1) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{{e_D}^2(x, y, n+1)}{2\sigma^2}\right) \tag{28}$$

$$\begin{aligned} {\nabla_n}^x = {\nabla_n}^y = \frac{1}{2\sigma^2} e_R(x, y, n+1) e_D(x, y, n+1) \\ \left(\tilde{f}(x, y, n) - \tilde{f}(x, y, n-1)\right) \end{aligned} \tag{29}$$

In this section, we have derived an instantaneous gradient estimator for the information potential following a methodology similar to the LMS algorithm. The only difference is that the training data set, now consists of pair-wise differences of the samples in contrast to using the actual input-output pairs as done in LMS. Here we can observe that the updates for both the motion vectors have the same value. This implies that the possible set of motion vectors is limited. Therefore to mitigate this problem, we impose a smoothness constraint that the neighboring motion vectors cannot differ by more than a pre-determined threshold. Thus, we have defined completely a method for adaptive motion estimation.

5 Complexity Analyses and Storage Requirements

We discuss the computational complexity and storage requirements of the adaptive motion estimation algorithm and compare it with EBMA, the block-based gradient descent search [7] algorithm and the three-step search [8], which will be presented in Sections 5.1 and 5.2, respectively.

5.1 Complexity Analyses

Assuming the image size is M x M, with a search range of R x R and a block size of B x B we proceed to calculate the number of operations required per pixel. Table 1 summarizes the number of additions, subtractions, exponentials, absolute values, multiplications and conditionals (if statements) required by the algorithm at the encoder [2].

Table 1
Comparison of worst case encoding complexity of the four schemes at the pixel level.

Operations per pixel	Adaptive Motion Estimation	EBMA	Block-based Gradient Descent	Three-Step Search
Additions	2	$(2R+1)^2$	$B^2+(R-2)(2B-1)$	$8(\log_2(R/2)+1)+1$
Subtractions	4	$(2R+1)^2$	$B^2+(R-2)(2B-1)$	$8(\log_2(R/2)+1)+1$
Exponentials	1	0	0	0
Absolute	1	0	0	0
Multiplications	4	0	0	0
Conditionals	4	$(2R+1)^2$	$B^2+(R-2)(2B-1)$	$8(\log_2(R/2)+1)+1$
Total	15	$4(2R+1)^2$	$4(B^2+(R-2)(2B-1))$	$4(8(\log_2(R/2)+1)+1)$

Table 2 shows the results for the case when R=16 and B=3, assuming all the above operations are executed in a single instruction cycle.

Table 2
Total number of instructions per pixel for R=16 and B=3.

Method	Total number of instructions per pixel
Adaptive Motion Estimation	15
EBMA	4356
Block-based Gradient Descent	316
Three Step Search	132

Our assumption that the above operations can be executed in a single instruction cycle is justified because the current DSP and FPGA technology can perform all the operations, except the exponential operation, in a single cycle. The exponential operation can be efficiently implemented using an 8-bit table lookup which also can be executed in a single instruction cycle. We have ignored assignment operations in the above calculations, as it is implementation dependent.

The above results show that our algorithm has extremely low computational complexity. It is nearly 220 times faster than EBMA, 21 times faster than Gradient descent and nearly 9 times faster than the three-step search (TSS). This means that the algorithm has great potential for use in encoders with limited computational capability like a wireless video sensor.

In Table 3, we present the decoding complexity of the various schemes and Table 4 shows the results for the case when R=16 and B=3, assuming all the above operations are executed in single instruction cycle.

Table 3
Comparison of worst case decoding complexity of the four schemes at the pixel level.

Operations per pixel	Adaptive Motion Estimation	EBMA	Block-based Gradient Descent	Three-Step Search
Additions	2	2	2	2
Subtractions	4	0	0	0
Exponentials	1	0	0	0
Absolute	1	0	0	0
Multiplications	4	0	0	0
Conditionals	4	0	0	0
Total	15	2	2	2

Table 4
Total number of instructions per pixel for R=16 and B=3.

Method	Total number of instructions per pixel
Adaptive Motion Estimation	15
EBMA	2
Block-based Gradient Descent	2
Three Step Search	2

The table above shows that for our scheme, the decoder complexity is the same as that of the encoder as expected. The decoder has 7 times more complexity than other schemes which have only 2 operations per pixel. However, the advantage of our scheme is that it does not require a separate hardware for the decoder because we start with the same initial conditions. Only the data input to the decoder is different. Also the increased complexity is minimal.

5.2 Storage requirements

The update equation for adaptive motion estimation requires storage of the current and the past motion compensated frames, the motion vectors for the previous frame, and the current and the past error frames. Also we need an 8 bit lookup table for exponent calculations. For other three algorithms, we require the storage of only 1 past frame. For wireless applications, computational complexity has higher precedence over storage requirement. In a typical wireless sensor network, a Y-QCIF video sequence would require approximately 175 kB of storage which is acceptable for most wireless applications.

Remark 1 One of the potential applications of our scheme is wireless video sensor networks. A wireless video sensor only employs video encoding but not video decoding. This requirement leverages the features of our algorithm, i.e., very low encoding complexity and slightly high decoding complexity. Hence, our algorithm is well suited for wireless video sensor networks because a video sensor is battery operated and hence power efficiency is especially critical.

6 Simulation results

In this section, we implement our adaptive motion estimation algorithm as described in Section 4. We choose the luminance component of several video sequences in QCIF format for the encoding process. For EBMA, a block size of 8x8 is chosen with integer-pel accuracy. The search range is 16x16 pixels. The block-based gradient descent search algorithm was implemented as described in [7] with a block size of 3x3 and a search range of 16x16 pixels with integer-pel accuracy. For the three-step algorithm [8] we use a block size of 8x8 and a search range of 16x16 pixels with integer-pel accuracy. The mean absolute error (MAE) distortion function is used as the block distortion measure for the two algorithms. Since we focus on the study of motion estimation, we have excluded DCT, quantization and entropy coding in the simulation.

In each algorithm, motion is estimated and compensated using the perfectly reconstructed reference frames. The first frame is intra-coded and the rest, inter-coded. The experiment was conducted using frame rates of 10, 5 and 2 respectively. The values of Y-PSNR in dB for the four different QCIF sequences are shown in Tables 5, 6 and 7.

Table 5
Y-PSNR values for 4 test video sequences at 10 fps.

Method	Miss America	Coastguard	Suzie	Foreman
EBMA	38.93	28.77	34.3	29.9
Three-step	35.96	26.45	28.51	23.92
Gradient Descent	31.5	25.04	23.9	20.64
Adaptive Motion Estimation	32.3	21.83	26.2	21.72

For low bit rate applications, the typical frame rate is usually 10 frames/sec or lower. As frame rate decreases, the temporal correlation between two consecutive video frames decreases. The assumption that block motion vectors are center-biased i.e. smaller displacements is more probable than larger ones,

Table 6
Y-PSNR values for 4 test video sequences at 5 fps.

Method	Miss America	Coastguard	Suzie	Foreman
EBMA	38.13	27.16	32.07	27.3
Three-step	33.68	22.49	25.9	21.3
Gradient Descent	29.53	23.01	22.41	19.02
Adaptive Motion Estimation	29.19	20.19	23.18	18.84

Table 7
Y-PSNR values for 4 test video sequences at 2 fps.

Method	Miss America	Coastguard	Suzie	Foreman
EBMA	36.93	25.03	29.35	23.07
Three-step	28.39	20.75	23.03	18.05
Gradient Descent	26.26	21.21	20.49	16.59
Adaptive Motion Estimation	25.81	18.66	20.20	16.18

fails here. This decreases the efficiency of block matching algorithms resulting in bigger motion vectors. If the search range is not big enough higher error values result. This increases the coded error value and the value of the motion vectors which are both undesirable for low bit rate coding. Our scheme is a frame based scheme so we can ideally find the true motion vector. We are not constrained by a search window. More the skip rate, the smaller is the probability of finding the true motion vector. The true motion vector can fall outside the search range. Transmission of motion vectors requires a significant amount of bits per pixel (bpp). Our scheme does not suffer from this because the motion vectors need not be transmitted. Thus, the adaptive motion estimation provides a trade-off between computational complexity and video presentation quality.

From Table 5 to Table 7, we notice that there is a 3 dB difference in Y-PSNR values between our algorithm and the three-step search. However, our scheme saves the bit budget for motion vectors, which usually constitutes about 50% of the total budget for low bit-rate video applications. Therefore, the 3 dB performance loss can be compensated by the bandwidth savings due to not transmitting motion vectors in our scheme.

Remark 2 In the simulator, we do not implement DCT, quantization and entropy coding. This is because the performance of a motion estimation algorithm is determined by the MSE of the error image, which is independent of DCT, quantization and entropy coding. From the information theory, it is known that typically, the larger the MSE of the error image without DCT, quantization and entropy coding, the larger MSE of the error image with DCT, quantization and entropy coding.

7 Conclusion

Motion estimation is a critical problem in the design of a video encoder. Existing motion estimation techniques do not effectively utilize the past knowledge in motion prediction, leading to inefficiency in computation. To address this problem, we proposed an adaptive model-based motion estimation algorithm using Renyi's Entropy. In this scheme, the motion vectors of the current frame are iteratively computed from the previous frame, based on a model. This results in computational savings because of the knowledge gained in the computation of the previous motion vectors, and it also leads to bandwidth savings because the motion vectors need not be transmitted. Our results showed that our scheme reduces the computational complexity by a factor of nine to twenty-one, as compared to the existing fast algorithms.

The nice feature of our adaptive motion estimation algorithm is its very low computational complexity. Hence, our algorithm is ideally suited for wireless video applications, in which computational complexity and energy consumption pose major constraints. With the emergence of wireless video sensor networks, we expect that our algorithm will find widespread applications.

Our future work will focus on implementing our algorithm on MPEG-4 and H.264 codecs and evaluating the performance of the resulting codecs. We will also investigate the error robustness of our algorithm. In addition, we will combine our algorithm with wavelet video coding.

References

[1] D. Erdogmus, J. Principe, Comparison of entropy and mean square error criteria in adaptive system training using higher order, in: Proc. Intl. Workshop on Independent Component Analysis and Signal Separation, Helsinki, Finland, 2000, pp. 75–80.

[2] Y. Wang, J. Ostermann, Y. Zhang, Video Processing and Communications, Prentice Hall, 2001.

[3] C. Stiller, J. Konrad, Estimating motion in image sequences, IEEE Signal Processing Magazine 16 (4) (1999) 70–91.

[4] M. E. Al-Mualla, C. N. Canagarajah, D. R. Bull, Video Coding for Mobile Communications: Efficiency, Complexity and Resilience, Signal Processing and Its Applications, Academic Press, San Diego, CA, 2002.

[5] ISO/IEC IS 10918-1 (ITU-T Rec.T.81), Digital compression and coding of continuous-tone still images; part 1:requirements and guidelines (Feb. 1994).

[6] J. R. Jain, A. K. Jain, Displacement measurement and its application in interframe image coding, IEEE Trans. Commun. 29 (12) (1981) 1799–1808.

[7] L. Lurng-Kuo, E. Feig, A block-based gradient descent search algorithm for block motion estimation in video coding, IEEE Trans. Circuits Syst. Video Technol. 6 (4) (1996) 419–422.

[8] T. Koga, K. Linuma, A. Hirano, Y. Iijima, T. Ishiguro, Motion-compensated inter-frame coding for video conferencing, in: Proc. NTC, New Orleans, LA, 1981, pp. G5.3.1–5.

[9] R. Li, B. Zeng, M. L. Liou, A new three-step search algorithm for block motion estimation, IEEE Trans. Circuits Syst. Video Technol. 4 (4) (1994) 438–442.

[10] M. P. Lai, W. C. Ma, A novel four-step search algorithm for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol. 6 (3) (1996) 313–317.

[11] B. Liu, A. Zaccarin, New fast algorithms of the estimation of block motion vectors, IEEE Trans. Circuits Syst. Video Technol. 3 (2) (1993) 148–157.

[12] K. M. Nam, J. S. Kim, R. H. Park, Y. S. Shim, A fast hierarchical motion vector estimation algorithm using mean pyramid, IEEE Trans. Circuits Syst. Video Technol. 5 (4) (1995) 344–351.

[13] M. Bierling, Displacement estimation by hierarchical block matching, in: SPIE Visual Communications and Image Processing, Vol. 1001, 1998, pp. 942–951.

[14] B. C. Song, J. B. Ra, A hierarchical block matching algorithm using partial distortion criteria, in: SPIE Visual Communications and Image Processing, 1998, pp. 88–95.

[15] W. Li, E. Salari, Successive elimination algorithm for motion estimation, IEEE Trans. Image Processing 4 (1) (1995) 105–107.

[16] Y. S. Chen, Y. P. Hung, C. S. Fuh, Fast block matching algorithm based on the winner-update strategy, IEEE Trans. Image Processing 10 (8) (2001) 1212–1222.

[17] J. Yeh, M. Khansari, M. Vetterli, Motion compensation of motion vectors, in: Proc. of IEEE International Conference on Image Processing, Vol. 1, 1995, pp. 574–577.

[18] A. Renyi, Probability Theory, American Elsevier Publishing Company Inc., 1970.

[19] D. Erdogmus, J. C. Principe, An on-line adaptation algorithm for adaptive system training with minimum error entropy: Stochastic information gradient, in: Proc. Intl. Workshop on Independent Component Analysis and Signal Separation, San Diego, CA, 2001, pp. 7–12.

[20] S. Haykin, Adaptive Filter Theory, 4th Edition, Prentice-Hall, 2001.

[21] D. Erdogmus, J. C. Principe, Generalized information potential criterion for adaptive system training, IEEE Trans. Neural Networks 13 (5) (2002) 1035–1044.

[22] E. Parzen, On estimation of a probability density function and mode, in: Time Series Analysis Papers, Holden-Day, San Francisco, CA, 1967.

Ganesan Ramachandran was born in Madurai, India, on February 10th, 1978. Ganesan was awarded a Bachelor of Engineering degree in Electronics and Communication Engineering in 1999 from Regional Engineering College, Tiruchirappalli, India. After finishing his undergraduate degree, Ganesan worked as a VLSI and Embedded Systems Engineer at WIPRO Technologies Ltd., Bangalore, India from 1999 to 2000. In fall 2000, Ganesan enrolled at the University of Florida to start his graduate study in Electrical and Computer Engineering. In summer 2002, under Dr. Jose Principe's supervision, Ganesan completed his M.S. degree with his research in combining traditional signal processing methods with nonlinear dynamic methods for epilepsy research. He is presently pursuing his Ph.D. degree under the supervision of Dr. Paul D. Gader specializing in neural networks and signal processing for landmine detection.

Vignesh Krishnan studied electronics and communication engineering at Coimbatore Institute of Technology, Coimbatore, India from 1998-2002. In Fall 2004, Vignesh will be receiving a Masters degree in Electrical Engineering at the University of Florida. He will then be pursuing his Ph.D. under the supervision of Dr. Dapeng Wu at the same University. His research interests include video compression, video communications, signal processing, networking and multimedia. He is a student member of the IEEE.

Dapeng Wu received B.E. in Electrical Engineering from Huazhong University of Science and Technology, Wuhan, China, in 1990, M.E. in Electrical Engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 1997, and Ph.D. in Electrical and Computer Engineering from Carnegie Mellon University, Pittsburgh, PA, in 2003. From July 1997 to December 1999, he conducted graduate research at Polytechnic University, Brooklyn, New York. During the summers of 1998, 1999 and 2000, he conducted research at Fujitsu Laboratories of America, Sunnyvale, California, on architectures and traffic management algorithms in the Internet and wireless networks for multimedia applications.

Since August 2003, he has been with Electrical and Computer Engineering Department at University of Florida, Gainesville, FL, as an Assistant Professor. His research interests are in the areas of networking, communications, multimedia, signal processing, and information and network security. He received the IEEE Circuits and Systems for Video Technology (CSVT) Transactions Best Paper Award for Year 2001.

Currently, he is an Associate Editor for the IEEE Transactions on Vehicular Technology and Associate Editor for International Journal of Ad Hoc and Ubiquitous Computing. He served as Program Chair for IEEE/ACM First International Workshop on Broadband Wireless Services and Applications (BroadWISE 2004); and as TPC member of over 20 conferences such as IEEE INFOCOM'05, IEEE ICC'05, IEEE WCNC'05, and IEEE Globecom'04. He is Vice Chair of Mobile and wireless multimedia Interest Group (MobIG), Technical Committee on Multimedia Communications, IEEE Communications Society. He is a member of the Award Committee, Technical Committee on Multimedia Communications, IEEE Communications Society. He is also Director of Communications, IEEE Gainesville Section.

Zhihai He received the B.S. degree from Beijing Normal University, Beijing, China, and the M.S. degree from Institute of Computational Mathematics, Chinese Academy of Sciences, Beijing, China, in 1994 and 1997 respectively, both in mathematics, and the Ph.D. degree from University of California, Santa Barbara, CA, in 2001, in electrical engineering. In 2001, he joined Sarnoff Corporation, Princeton, NJ, as a Member of Technical Staff. In 2003, he joined the Department of Electrical and Computer Engineering, University of Missouri, Columbia, as an assistant professor. He received the 2002 IEEE Transactions on Circuits and Systems for Video Technology Best Paper Award, and the SPIE VCIP Young Investigator Award in 2004. His current research interests include image/video processing and compression, network transmission, wireless communication, computer vision analysis, sensor network, and embedded system design. He is a member of the Visual Signal Processing and Communication Technical Committee of the IEEE Circuits and Systems Society, and serves as Technical Program Committee member or session chair of several international conferences.