A Novel Vector Quantizer-Based Architecture for Block-Matching Motion Estimation

Ali Bilgin Hüseyin Abut† Michael W. Marcellin

Department of Electrical & Computer Eng.
University of Arizona
Tucson, AZ 85721

†Department of Electrical & Computer Eng.
San Diego State University
San Diego, CA 92182

Abstract

Block-matching algorithms (BMA) are commonly used in image sequence coding for motion estimation. Even though the estimation of block motion vectors by an exhaustive search is optimal, the computational complexity precludes its usage in many real-time codec applications. In this paper, we present a novel vector quantizer (VQ) based architecture for BMA offering sizeable reduction in the number of computations and also the number of motion vectors transmitted. We also compare the performance of the proposed algorithm with other commonly used fast search methods.

1. INTRODUCTION

Among the most frequently used motion estimation techniques for image sequence coding are the block-matching algorithms (BMAs). Even though a block matching estimation using exhaustive search can result in an optimal solution with respect to an adopted fidelity criterion, the computational complexity associated with a full search BMA precludes its usage in many applications. In the literature, many fast BMA's have been proposed to reduce the computational complexity [2, 3, 4, 5].

In this paper, we propose a Finite-State Vector Quantizer (FSVQ) based architecture for estimating the block motion vectors. The proposed architecture is compatible with the current MPEG II [1] video standard. It is quite comparable to previously proposed fast techniques in terms of entropy reduction performance. In addition, it offers vastly reduced computational requirements and produces a smaller number of motion vectors to transmit. Finally, the proposed algorithm also offers the flexibility to adaptively change the codebook size of the VQ to trade between the computational complexity and the entropy of the motion compensated residue.

2. BLOCK-MATCHING ALGORITHM

In a block-matching algorithm, an image frame is divided into blocks of $M \times M$ picture elements (pels). In an optimal exhaustive search BMA with a maximum motion displacement of W pels, $(2W+1)^2$ motion vectors are checked for every block and a distortion corresponding to each motion vector is computed based on a block distortion measure (BDM). The best match with respect to this BDM yields the motion vector for the block.

Since block matching estimation using exhaustive search requires $(2W+1)^2$ computations of the BDM, it is often unrealizable in real-time due to computational complexity. To reduce this complexity, several fast algorithms have been proposed, including Modified Logarithmic Search (MLS) [2], Conjugate Directions Search (CDS) [4], and Three-Step Search (TSS) [3]. All of these algorithms perform some systematic search over the search region to determine a vector that approximately minimizes the BDM.

In many real world image sequences, large segments of an image frame are stationary and the motion in the rest of the frame is due to the motion of only a few objects. This implies that the motion field of most frames in a sequence is rather stationary. Since a BMA attempts to represent smooth motion fields with a discrete number of block motion vectors, the correlation among motion vectors is usually high [6]. Moreover, in many applications, motion vectors are highly correlated due to the motion in only one direction. In Figure 1 we present the motion vector distribution of a typical image sequence of 150 frames. As seen in this figure, the distribution is not uniform. On the contrary, the majority of the motion vectors lie inside a very small region. Thus, it might be possible to achieve significant compression in the form of an entropy reduction by considering only a small subset of motion vector candidates.

1058-6393/96 $5.00 © 1996 IEEE
Proceedings of ASILOMAR-29

Recently, Lee and Woods reinterpretted the BMA as a type of vector quantization called motion vector quantization (MVQ) [5]. Their algorithm simultaneously estimates and vector quantizes motion vectors by minimizing the variance of the displaced frame difference (DFD). Since the fidelity criterion employed is significantly different from the ones used in the other papers mentioned above, it would be very difficult to assess the performance of this last study. Hence, we will compare our findings in this study to those of the modified logarithmic search and the conjugate directions search techniques.

3. VECTOR QUANTIZATION FOR BMA

As Gersho and Gray recently stated in [7], vector quantization (VQ) has proved to be a valuable coding technique in a variety of applications due to its simplicity and its ability to trade memory for expensive computation among other promises. Here we attempt to benefit from these advantages of VQ to reduce the computational complexity of block matching algorithms used in motion estimation. If the maximum displacement allowed in either direction is W, then $(2W+1)^2$ possible candidates exist for the motion vector. Using the VQ design techniques based on the Generalized Lloyd Algorithm (GLA), if a codebook with only N codewords is designed for these $(2W+1)^2$ possible motion vectors, where $N \ll (2W+1)^2$, then the number of computations can be reduced significantly. Furthermore, the amount of side information required to transmit the motion vectors from the encoder to the decoder is reduced proportionally by the ratio: $\log_2(N)/\log_2((2W+1)^2)$.

3.1. Codebook Design

Fast search block matching algorithms are based on the assumption that the energy of the prediction error is monotonically decreasing towards an optimum motion vector in the search range. A VQ can be designed to exploit this property.

Since the design and encoding stages of full search vector quantizers require very large computational capacity, several alternative approaches have been proposed [7, 8]. One such approach for designing a VQ codebook, namely greedy tree growing algorithm, was introduced in [9]. Here Makhoul et al. suggested that a single terminal node of a full binary tree could be grown at a time instead of growing a level of the tree at a time. In the center of this technique lies the idea that the node contributing most to the overall distortion is selected and split. The advantage of this scheme is that more codewords are made available to code the high distortion events. This algorithm was further improved by Riskin [10] through modifications of the splitting criterion. The splitting criterion in Riskin's algorithm is to split the node which will yield the largest decrease in distortion-rate ratio rather than to split the node with the largest distortion. Riskin's algorithm was used for implementing variable rate VQ coders. Since in this work we will be using fixed rate VQ architectures, we will use a slightly modified version of her algorithm which is presented below in pseudocode. Here, $\Delta d(t)$ is the decrease in distortion by splitting node t into a left child t_L and a right child t_R. It is computed by

$$\Delta d(t) = \sum_{x \in \tau} d(x,t) - \sum_{x \in \tau_L} d(x,t_L) - \sum_{x \in \tau_R} d(x,t_R)$$

where τ is the group of training vectors belonging to the codeword t. `CodebookSize` is the current number of codewords in the codebook. `N` is the required size of the codebook. `DELTALIST` is the list of $\Delta d(t)$ values for all the codewords.

The Algorithm:

```
Compute centroid of the training data;
Split this node into t_L and t_R using GLA;
Insert t_L and t_R into the codebook;
Set CodebookSize = 2;
while (CodebookSize < N ) {
    Split t_L using GLA and calculate Δd(t_L);
    Split t_R using GLA and calculate Δd(t_R);
    Insert Δd(t_L) and Δd(t_R) into DELTALIST;
    Search DELTALIST for Δd_max and find
       the corresponding t_max;
    Delete Δd_max from DELTALIST and t_max
       from the codebook;
    t_L = t_max's left child;
    t_R = t_max's right child;
    Insert t_L and t_R into the codebook;
    CodebookSize = CodebookSize +1;
}
```

For BMA, the training set consists of motion vectors which have been previously computed using the exhaustive search. Using this training set, an unbalanced greedy tree is grown until N terminal nodes are obtained and these N terminal nodes are the motion vector codewords [1]. In designing the VQ, one can use the MSE fidelity criterion or the BDM which will be used later in the estimation stage. In this work, we have used the ubiquitous MSE during the design, but the MAD was used as the BDM during the encoding stage to expedite the search process.

[1]This stopping criterion, —Stop after N codewords— is the major difference between the current study and that of Riskin [10].

Once the motion vector codebook is designed, motion estimation is performed by computing a block distortion value for every codeword in the codebook. Then the codeword with the minimum distortion is selected as the motion vector. This technique requires N searches over the search range. N is expected to be rather small if a high degree of correlation exists among the motion vectors, which has been the case in most studies and our limited experiments have also verified this.

4. FINITE- STATE VQ FOR BMA

It is frequently observed that the motion fields of successive frames exhibit a high degree of short-term interframe correlation in addition to the long-term interframe and intraframe correlations as discussed above. In other words, since the motion field does not usually change rapidly, the motion vectors of corresponding blocks in successive frames are highly correlated. Thus, it is reasonable to assume that the corresponding blocks of successive frames have similar motion vectors and they will be in neighboring Voronoi regions in the VQ framework. We will put this property to use in the next paragraph to further improve the VQ codebook design.

Using the above unbalanced greedy tree growing algorithm, a VQ of codebook size N is designed with respect to the MSE distortion measure. For every codeword of this "supercodebook", $K-1$ nearest neighbor codewords are found to form an N-state finite machine. In other words, for every codeword, $K-1$ additional state codewords are selected from the supercodebook of size N as depicted in Figure 2.

The motion estimation is performed by using the codewords in the state codebook specified by the motion vector of the corresponding block in the previous frame. In other words, if the motion vector of the corresponding block in the previous frame was *Codeword* M , then only the blocks corresponding to motion vectors $y_M^i, i = 1, ..., K-1$ and *Codeword* M are used in the estimation process. The codeword yielding the minimum distortion according to our BDM is selected as the motion vector. Due to the inherent suboptimality of this architecture, as in all other partial search techniques, it is possible to obtain inferior results during rapidly changing scenes. To overcome this problem, the distortion of the selected motion vector is compared to an experimentally determined distortion threshold. This modification can be interpreted as a form of distortion constrained VQ. If the codeword distortion is greater than a threshold, a full search is conducted over the entire supercodebook and the codeword yielding the minimum distortion is chosen to be the motion vector.

Another advantage of this architecture is its ability to adapt to changing system resources. For example, an architecture can be created by designing the state codebooks to include every codeword in the supercodebook in descending nearest-neighbor order. Depending on the available computational power and/or channel capacity of the system, only a certain number of these codewords are used in the estimation process. If the system status changes, the number of codewords used in estimation can be increased or decreased to adapt to the new requirements. The block diagram of such a FSVQ-based BMA is illustrated in Figure 3. Here, the next-state function simply retrieves the motion vector of the corresponding block in the previous frame from the motion vector store and sets the next state to this vector.

In addition to reducing the computational complexity, our FSVQ-based BMA also reduces the amount of information required to represent the motion vectors. Since in BMA the motion vectors are transmitted as side information, the number of motion vector candidates affects the amount of information to be transmitted. For large search regions, this information can get quite large. Since VQ-based motion estimation limits the number of motion vector candidates, the amount of side information to be transmitted is lowered considerably as presented in the next section.

5. EXPERIMENTAL RESULTS

We have performed motion estimation using forward prediction with a block size of 16×16 on the 8-bit luminance component (Y) of the Miss America sequence. In this sequence there are 150 frames each with 680×480 pels. First, 90 frames of this sequence were chosen for training and the remaining 60 were used during the testing phase. The maximum horizontal and vertical displacements were chosen to be six pels as in prior studies. The MAD was adopted as the BDM, which resulted in estimates similar to those of more complex measures.

5.1. Entropy of the Residual Signal

Since the goal of a motion compensator is to reduce the information content of the residual signal, the entropy of the residual signal has been commonly chosen as the fidelity criterion for motion estimation algorithms. We have adopted that approach for the work presented here.

Before motion estimation is performed, we apply the motion detection algorithm presented by Puri et al. [11] to the Miss America sequence. This algorithm first classifies every pixel of a particular block as 'moving' or 'non-moving' by comparing the intensity differ-

ence between the current pixel and the corresponding pixel in a reference frame, to a predetermined threshold. Next, the number of 'moving pixels' in this block are determined. If the number of 'moving pixels' exceeds a threshold, the block is considered to be 'moving'. In our experiments, we have found the algorithm to work adequately for $T_o = 5$ and $N_o = 50$ for blocks of size 16×16 and 8-bit gray level pixels, where T_o represents the intensity difference threshold and N_o is the threshold for the number of varying pixels in a block. With this algorithm we have classified around 60%–70% of the blocks as 'non-moving' with only 0.1 bits/pixels increase in entropy. [2]

We present the comparison of our VQ based algorithms with conventional algorithms in Figure 4. It can be seen that the full search VQ based algorithm with a codebook size of 10 codewords almost always outperforms the conventional fast algorithms mentioned earlier. The codebooks needed in our FSVQ-based algorithm were designed with a supercodebook size of 20 codewords, a state codebook size of 5 codewords, and a distortion threshold of 1600. This algorithm also performs at least as well or better than the conventional algorithms.

5.2. Computational Requirements

Figure 5 shows the computational requirements of the fast motion estimation algorithms. As seen in that figure, FSVQ-based BMA requires about half the number of BDM evaluations of conventional fast algorithms. It should also be noted that the exhaustive search BMA requires around 60,000 block distortion computations per frame, which are not included. Those results have not been included to allow closer inspection of the performance of the fast algorithms.

5.3. Reduction in the Motion Vector Data Rate

To numerically illustrate the savings in the data transmission rate for motion vectors, we consider the case with a maximum horizontal and vertical displacement of 6 pels yielding a total of 169 candidates for motion vectors. This would require 8 bits per motion vector to be transmitted as side information. For a block size of 16×16 and a frame size of 680×480 pels at 30 frames per second, this would amount to 288 kbits/s. However as illustrated in the previous section for the same image size, block size, and frame rate, a FSVQ-based motion estimation algorithm performs comparably to the exhaustive search using only 20 candidates thus, requiring only 5 bits per motion vector. [3] This corresponds to only 180 kbits/s of side information, which is equivalent to a 37.5 percent savings in this side information data rate.

6. CONCLUSIONS

In this paper, a new approach for fast BMA has been introduced. Unlike the previously proposed fast BMAs, we have approached the task of reducing the search locations in BMA as a VQ problem and limited the search using a VQ. The result is a new FSVQ-based BMA. The proposed algorithm has yielded extremely encouraging results in our experiments. It has usually outperformed the conventional fast algorithms in terms of entropy reduction. More impressively, it requires only half as many computations as the conventional fast algorithms. Additionally, this algorithm requires 37.5 % less side information to be transmitted for motion vectors. The proposed algorithm also offers adaptivity to trade between the computational complexity and the entropy of the motion compensated residue.

References

[1] MPEG, "Generic Coding of Moving Pictures and Associated Video ", Tech. Rep., CD 13818-2, ISO/IEC JTC1/SC29, Nov. 1993.

[2] S. Kappagantula and K. R. Rao, "Motion predictive interframe coding", *IEEE Transactions on Communications*, vol. 33, pp. 1011–1015, Sept. 1985.

[3] K. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing", in *National Telecommunications Conference*, Nov. 1981, pp. G5.3.1–G5.3.5.

[4] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation", *IEEE Transactions on Communications*, vol. 33, pp. 888–896, Aug. 1985.

[5] Y. Y. Lee and J. W. Woods, "Motion vector quantization for video coding", *IEEE Transactions on Image Processing*, vol. 4, pp. 378–382, Mar. 1995.

[6] Michael T. Orchard, "A comparison of techniques for estimating block motion in image sequence coding", in *SPIE, Visual Communications and Image Processing IV*, 1989, vol. 1199, pp. 248–258.

[2] Only the motion vectors from blocks classified as moving in the first 90 frames of the Miss America sequence were used as training data for VQ codebook design.

[3] These 5 bits could further be reduced to 3 by exploiting the state structure, but not within the scope of the MPEG II [1] standard.

[7] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Massachusetts, 1992.

[8] H. Abut, *Vector Quantization*, IEEE Press, New York, 1990.

[9] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding", *Proceedings of the IEEE*, vol. 73, pp. 1551–1588, 1985.

[10] E. A. Riskin, *Variable Rate Vector Quantization of Images*, PhD thesis, Stanford University, June 1990.

[11] A. Puri, H. M. Hang, and D. L. Schilling, "An efficient block matching algorithm for motion compensated coding", in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Apr. 1987, vol. 2, pp. 25.4.1–25.4.4.

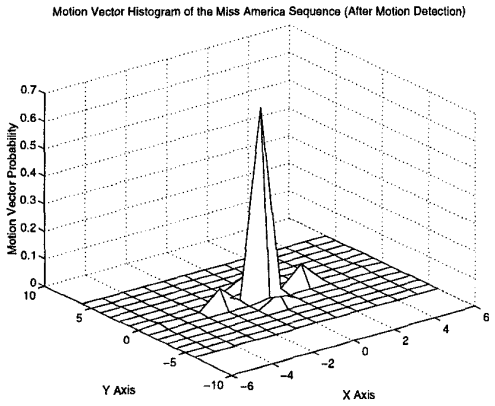

Figure 1: Motion Vector Distribution

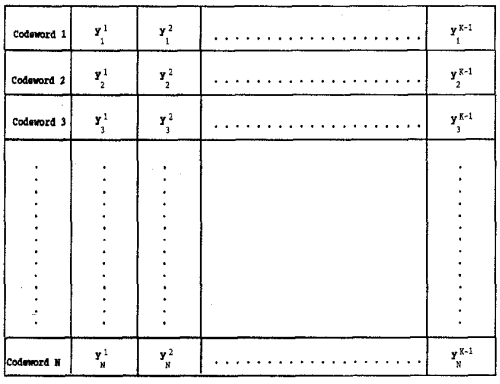

Figure 2: Finite State Vector Quantizer Codebook

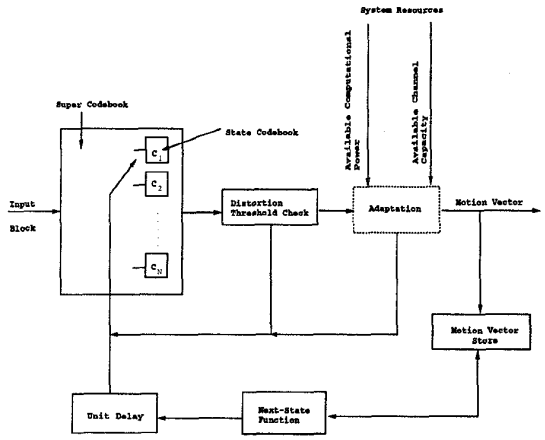

Figure 3: FSVQ-based BMA Encoder

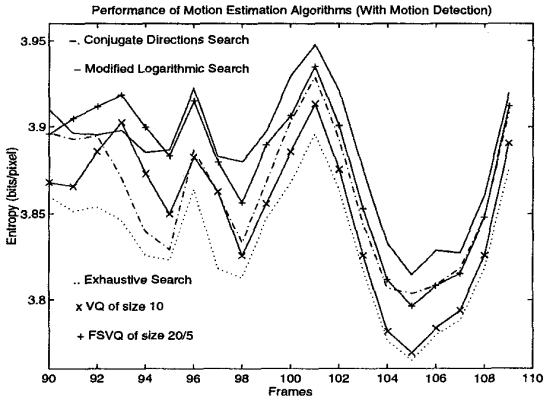

Figure 4: Comparison of the VQ-based Motion Estimation Algorithms with the Conventional Fast Motion Estimation Algorithms on the Miss America Sequence with Motion Detection (Frames 120-140)

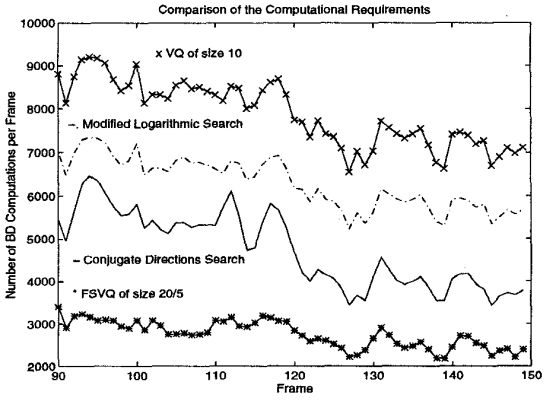

Figure 5: Computational Requirements of the Fast Motion Estimation Algorithms