Interframe Difference Quadtree Edge-Based Side-Match Finite-State Classified Vector Quantization for Image Sequence Coding

Ruey-Feng Chang, *Member, IEEE,* and Wei-Ming Chen, *Member, IEEE*

***Abstract*— Very low bit rate image sequence coding is very important for video transmission and storage applications. The fundamental goal of image sequence coding is to remove spatial redundancy and only update the moving parts in an image sequence so that the total information bits required for storage or transmission can be greatly reduced. In our proposed approach, each frame within an image sequence will be separated into moving and stationary blocks. Only the moving blocks need to be transmitted to the decoder, so that the total number of bits and computing time are greatly reduced. The moving blocks will be encoded by edge-based side-match finite-state classified vector quantization (EBSMCVQ). Moreover, a quadtree can be also used to represent the moving and edge information for each frame. In order to reduce the number of bits for the quadtree, we propose a new difference quadtree technique that only transmits the different parts between the previous frame's quadtree and the current frame's quadtree. In the proposed interframe difference quadtree EBSMCVQ image sequence coding scheme, the average bit rate of each frame is reduced to 0.0393 b/pixel and the PSNR is still up to 35.40 dB for image sequence Claire.**

I. Introduction

VECTOR quantization (VQ) [1]–[9] has been found to be an efficient scheme for image compression. The image to be encoded is first partitioned into nonoverlapping rectangular blocks (vectors). A codebook is generated by using an iterative clustering algorithm such as K-means, or the generalized Lloyd clustering algorithm (LBG), proposed by Linde *et al.* [1]. The input vectors are then quantized to the closest codewords in the codebook. Image compression is achieved by using the indexes or labels of the codewords for storage and transmission.

Finite-state vector quantization (FSVQ) [10]–[15] has been proposed in order to exploit the correlations between the neighboring vectors to reduce the bit rate. FSVQ selects a smaller state codebook from a large master codebook by the current encoder's state. Hence, an FSVQ can maintain the image quality of a large VQ codebook, and achieve the bit rate efficiency of a small VQ codebook. Side-match VQ (SMVQ) [15] exploits the upper block's codeword and the left block's codeword for the current input vector to predict the currently encoded vector. SMVQ can select codewords that are very close to the original vector from the master codebook. There are some problems when using SMVQ under certain circumstances. If the correlations between neighboring vectors are not high, then SMVQ will select codewords that are not close to the input vector and increase quantization distortion. Hence, Chang and Chen [16] proposed an edge-based side-match finite-state classified vector quantizer (EBSMCVQ) that exploits classified VQ (CVQ) [17] to improve the SMVQ. In CVQ, the blocks are classified into several classes. Each class has a different codebook, and this codebook is generated according to the characteristics of the class.

EBSMCVQ can also be applied to image sequence coding. First, each block is classified into a moving or stationary block. Only the moving blocks are transmitted to the decoder thus reducing the bit rate. The moving blocks are encoded by the EBSMCVQ. The bit rate for the moving and edge information can be reduced by the quadtree scheme. In order to further reduce the bit rate for moving and edge information, we propose a new difference quadtree technique. The decoder will use a difference quadtree to obtain the current frame's quadtree from the previous frame's quadtree.

The organization of this paper is as follows. In Section II, we briefly review the main features of VQ, CVQ, FSVQ, SMVQ, and EBSMCVQ. The interframe difference quadtree EBSMCVQ is proposed in Section III. Section IV presents the experimental results. Some conclusions are drawn in Section V.

II. Edge-Based Side-Match Finite-State Classified VQ with Quadtree Map

Vector quantization is defined as a mapping from a k-dimensional Euclidean space R^k to a finite subset of R^k. This finite set $C = \{\hat{x}_i: i = 1, \cdots, N\}$, where N is the size of the codebook, is called a VQ codebook. A complete VQ coder has two parts: an encoder that assigns each input vector $\boldsymbol{x} \in R^k$ to an index i and a decoder that finds the codeword from the transmitted index i. The distortion between the input vector $\boldsymbol{x}$ and its corresponding codeword $\hat{\boldsymbol{x}}$ is measured by the squared Euclidean distortion measure, i.e.,

$$d(\boldsymbol{x}, \hat{\boldsymbol{x}}) = ||\boldsymbol{x} - \hat{\boldsymbol{x}}||^2 = \sum_{j=0}^{k-1} (x_j - \hat{x}_j)^2. \tag{1}$$

Manuscript received July 4, 1994; revised March 28, 1995 and October 5, 1995. This paper was recommended by Associate Editor K.-H. Tzou. This work was supported by the National Science Council, Taiwan, Republic of China, under Grant NSC83-0404-E-194-026.

R.-F. Chang is with the Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 62107, R.O.C.

W.-M. Chen is with the Computer Center, National Dong Hwa University, Hualien, Taiwan 970, R.O.C.

Publisher Item Identifier S 1051-8215(96)01325-0.

Classified VQ selects a particular subset of the codebook instead of the original large codebook by one or more features from the input vector, such as the block mean, block energy, edge, and other statistics. Each subset of the codebook is called a class codebook.

A. Side-Match Finite-State Vector Quantization

An FSVQ uses the previously encoded blocks to establish a uniquely defined state. This state may be described by a state variable $s \in S = \{s_i: i = 1, \cdots, M\}$. Hence, an FSVQ can be defined as a mapping from $R^k \times S$ to a subset of a master codebook $C = \{\hat{\boldsymbol{x}}_i: i = 1, \cdots, N\}$. For each state s_i, we select N_f codewords from the master codebook C as the state codebook SC_s. For encoding an input vector $\boldsymbol{x}$, the encoder finds the current state s and then searches the state codebook SC_s, not the master codebook, to find its corresponding codeword. The decoder will find the same current state s and the corresponding codeword in the same state codeword SC_s from the received index.

Side match vector quantizers are a class of FSVQ. For the blocks in an image, there may be high statistical correlations between the neighboring blocks. The SMVQ tries to make the intensity (gray level) transition across the boundaries of the neighboring blocks as smooth as possible. Therefore, it uses the sides of upper and left neighboring blocks' codewords to generate the state codebook SC_s for the current input vector $\boldsymbol{x}$, as shown in Fig. 1. The state space S is defined as $C \times C = \{\boldsymbol{u} \times \boldsymbol{l} | \boldsymbol{u}$ is the upper block's codeword and $\boldsymbol{l}$ is the left block's codeword$\}$. The variable $\boldsymbol{u}$ can be used to keep the correlation of the neighboring blocks in the vertical direction, and the variable $\boldsymbol{l}$ can be used to keep that of the neighboring blocks in the horizontal direction. That is, the state s of an input vector $\boldsymbol{x}$ is defined by the column state $\boldsymbol{u}$ and the row state $\boldsymbol{l}$. The corresponding state codebook SC_s of the current input vector $\boldsymbol{x}$ is defined to be the codewords that best match the upper block and the left block to $\boldsymbol{x}$ along the neighboring columns and the neighboring rows, respectively. Let $\boldsymbol{u}$ be the upper block's codeword and $\boldsymbol{l}$ be the left block's codeword to the current input block. The horizontal side-match distortion of a codeword $\boldsymbol{y}$ in the master codebook C is defined as

$$hd(\boldsymbol{y}) = \sum_{j=1}^{n} (y_{1j} - u_{mj})^2 \tag{2}$$

and the vertical side-match distortion is defined as

$$vd(\boldsymbol{y}) = \sum_{i=1}^{m} (y_{i1} - l_{in})^2. \tag{3}$$

Further, the side-match distortion of a codeword $\boldsymbol{y}$ is defined as

$$smd(\boldsymbol{y}) = hd(\boldsymbol{y}) + vd(\boldsymbol{y}). \tag{4}$$

The selection algorithm for state s is to select N_f codewords with the smallest side-match distortion smd from C as the state codebook SC_s.

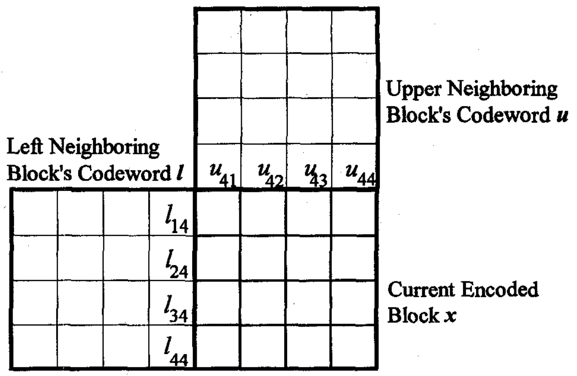

Upper Neighboring Block's Codeword u
Left Neighboring Block's Codeword l
u41
u42
u43
u44
l14
l24
l34
l44
Current Encoded Block x

Fig. 1. SMVQ encoding (block size = 4 × 4). Find the state s for the current block $\boldsymbol{x}$ by the codeword $\boldsymbol{u}$ of the upper block and the codeword $\boldsymbol{l}$ of the left block to $\boldsymbol{x}$.

B. Edge-Based Side-Match Finite-State Classified VQ with Quadtree Map

The main key problem of VQ is edge degradation. Although edge blocks may be only a small fraction of the image, the visual quality of the encoded image will be very annoying if the edge blocks are not encoded correctly. This situation occurs because the measure criterion MSE for selecting the closest codeword does not preserve edge information. Therefore, Ramamurthi and Gersho [17] proposed a classified VQ to avoid edge degradation. In their method, each block is categorized into several classes such as shade and edge blocks. Each class can use a different codebook and an appropriate encoding technique.

The side-match FSVQ algorithm exploits the high correlations between neighboring blocks to select a state codebook from the master codebook to reduce the bit rate. However, if the correlations are not high, a better approximation of the input vector may not be found in the state codebook. For a high variance image, there are no high correlations between neighboring blocks, so the ordinary SMVQ may reduce the image quality of the encoded image. For example, if an edge is across the lower right corner of a block, SMVQ will incorrectly encode the edge. The reason is that SMVQ uses only the left block and upper block to predict this block but the edge is across the right block and lower block. The EBSMCVQ algorithm combines SMVQ with CVQ. Not only neighboring blocks, but also the current block itself, is used to improve the image quality. In EBSMCVQ, the state codebooks are decided by not only the correlations between neighboring vectors but also the characteristics of the input vectors. A vector classifier is used to classify the input vectors into two classes, nonedge blocks and edge blocks. The edge information for each block can be reduced by using a quadtree data structure. First, all the nonedge blocks are encoded by an SMVQ with a smaller state codebook size. The edge blocks are further classified into 16 subclasses according to characteristics, edge or nonedge, of their neighboring blocks in order to improve the image quality in the edge areas. Each class of edge blocks is encoded by using a different master codebook in SMVQ. That is, there are 16 master codebooks for the edge blocks. In EBSMCVQ, all the nonedge blocks are encoded before the edge blocks. Hence, the side-match operation can use at most four neighboring, upper, left, lower, and right blocks to select codewords for the current state codebook. For example, in Fig. 2, the blocks $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ can be used to predict the currently encoded block $\boldsymbol{e}$ because they are encoded before block $\boldsymbol{e}$. Moreover, in EBSMCVQ, the size of the state codebook is variable. That size is dependant on the number of blocks that are encoded before the currently encoded edge block. If the number of the previously encoded neighboring blocks is larger, then the state codebook size could be smaller. Otherwise, the state codebook size for the currently encoded block will be larger.

EBSMCVQ transmits the quadtree map first, all the indexes of nonedge blocks, and then all the indexes of edge blocks. The advantage of the EBSMCVQ is that the bit rate for the low-detail blocks can be reduced and the image quality of the high-detail blocks can be improved. The reason is that the low-detail (low variance) blocks are encoded by using the smaller state codebook. Moreover, the encoded quality of the high-detail (edge) vectors can be improved by further classifying them into one of sixteen classes.

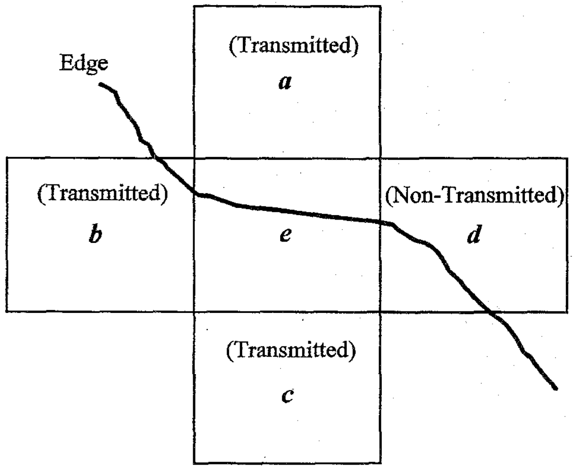


Fig. 2. Blocks $\boldsymbol{a}$ and $\boldsymbol{c}$ are nonedge blocks, blocks $\boldsymbol{b}$ and $\boldsymbol{d}$ are edge ones, and block $\boldsymbol{e}$ is the currently encoded edge block. Blocks $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ are transmitted before block $\boldsymbol{e}$ and block $\boldsymbol{d}$ is transmitted after block $\boldsymbol{e}$.

III. Interframe Difference Quadtree EBSMCVQ

In an image sequence, successive frames are usually very highly correlated. A simple frame replenishment coding system [4], [18], [19] transmits only the blocks that are changed between successive frames. Goldberg and Sun [18], [19] proposed the index replenishment VQ system. Only the indexes that are different from the previous indexes are transmitted to the decoder.

EBSMCVQ can also be applied to image sequence coding. First, each block in a frame is classified into a moving block or stationary block. Only the moving blocks are required to be encoded by EBSMCVQ and transmitted to the decoder in order to achieve very low bit rate coding.

A. Moving Blocks and Stationary Blocks

First, we describe the method for deciding whether a block $\boldsymbol{b}_n$ in the current frame n is a moving block or not. Let block $\boldsymbol{b}_{n-1}$ be the one at the same location as $\boldsymbol{b}_n$ in the previous frame $n-1$. Then, the class of the block $\boldsymbol{b}_n$ is decided by the edge types and the similarity of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$. If the edge types of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$ are different, then $\boldsymbol{b}_n$ should be updated. That is, $\boldsymbol{b}_n$ is a moving block. If the edge types of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$ are the same, then the moving type of $\boldsymbol{b}_n$ will be decided by the degree of similarity of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$. The degree of similarity is measured by the squared Euclidean distortion between $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$.

If the distortion of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$ is larger then a preset threshold TH_m, then block $\boldsymbol{b}_n$ is a moving one that should be updated in the current frame. Because edge blocks should be better encoded than nonedge blocks, two different TH_{me} and TH_{mn} values will be used to decide whether the block $\boldsymbol{b}_n$ is moving or stationary. That is, if $\boldsymbol{b}_n$ is an edge block, then a lower threshold TH_{me} can be used to decide whether $\boldsymbol{b}_n$ is moving.

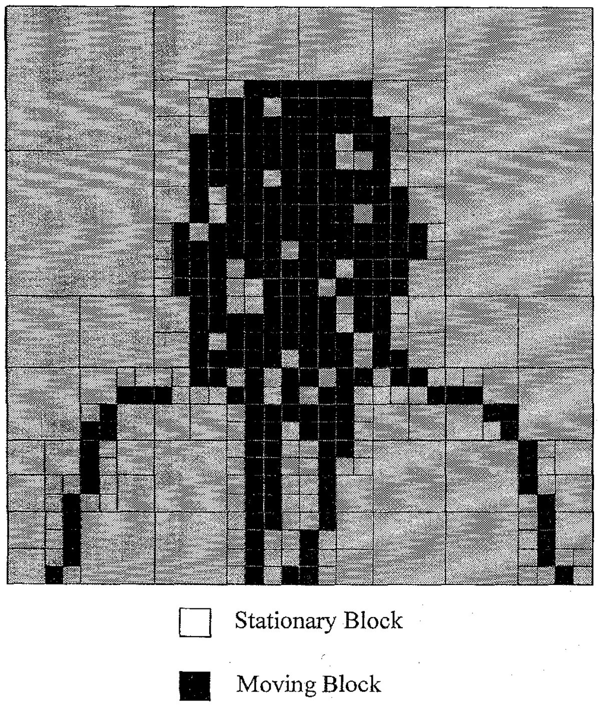


Fig. 3. The quadtree segmentation result of frame 7 in the test image sequence Claire.

Now, all the moving blocks that should be updated in the current frame can be encoded by EBSMCVQ. That is, the four neighboring blocks of $\boldsymbol{b}_n$ will be used to select the state codebook for the moving block $\boldsymbol{b}_n$. If block $\boldsymbol{b}_n$ is a nonedge block, then the state codebook will be selected from a nonedge master codebook. If block $\boldsymbol{b}_n$ is an edge block, the master codebook will be decided by the edge types of four neighboring blocks. Because the type of a block can be either edge or nonedge, the number of master codebooks is 16. Note that if a block is a moving block, then another side

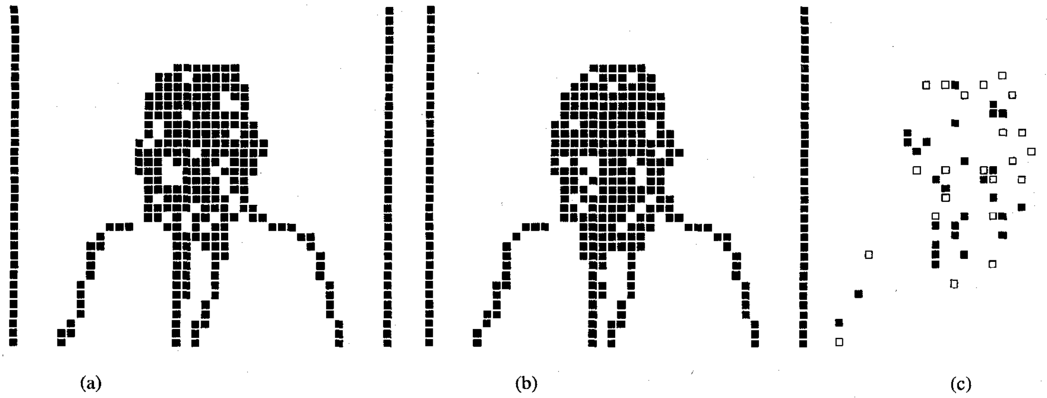


Fig. 4. (a) The moving information of frame 7 in the test image sequence Claire. (b) The moving information of frame 10. (c) The difference of the moving information between frame 7 and frame 10.

information bit is required to represent that this moving block is an edge or a nonedge block. For a stationary block, this side information is not needed because this stationary block and the previous frame's block at the same location have the same edge type. Moreover, the first frame of the image sequence has only edge information. The other successive frames will have both moving and edge information.

B. Quadtree for Moving and Edge Information

The quadtree algorithm is often used in image coding [20]–[25]. Strobach [20] proposed a tree-structured scene adaptive coder for image sequence coding. The motion compensated prediction (MCP) error images are encoded by the quadtree structured difference pulse code modulation (QSD-PCM). The sample mean of the block is used to decide whether this block is partitioned or not. Nasrabadi *et al.* [21] also proposed interframe hierarchical address-vector quantization for encoding image sequences. The MCP error image is partitioned into blocks of size 32×32, 16×16, 8×8, 4×4, and 2×2. If the variance of a larger block is larger than a threshold value, then the larger block is partitioned into four smaller blocks. The larger blocks (size 32×32, 16×16, and 8×8) are not transmitted to the decoder, and these blocks are replenished from the previous frame. However, the smaller blocks (size 4×4 and 2×2) are encoded by the address-vector quantization in order to faithfully reproduce edge details.

In the EBSMCVQ algorithm for encoding still images, the edge information for each block can be reduced by using a quadtree data structure. In the same way, the moving and edge information for each block in the interframe EBSMCVQ can also be reduced by using a quadtree data structure. The quadtree segmentation of the moving and edge information of frame 7 in the test image sequence Claire is shown in Fig. 3. Note that the quadtree is often used to segment the MCP error images, but our interframe EBSMCVQ uses the quadtree to represent the moving and edge information.

The quadtree segmentation is based on moving information instead of edge information. If there are any moving blocks in a larger region, then this region is partitioned into four smaller adjacent regions. The moving-based subdivision process is repeated until the region has no moving blocks. In general, because the moving region is much smaller, the type of larger regions is only stationary and the type of the smallest region can be moving or stationary. Note that the smallest region is a block in the quadtree segmentation. Because the type of each moving block can be edge or nonedge, an overhead bit for each moving block is needed.

C. Interframe Difference Quadtree EBSMCVQ

In the above quadtree EBSMCVQ, one quadtree is needed to represent the moving information for each frame. This overhead information should be reduced if this encoding scheme is applied to the very low bit rate coding. We can observe that the successive quadtrees Q_{n-1} and Q_n are very similar when the two successive frames n and $n-1$ are highly correlated. For example, the moving information of frame 7 and frame 10 in the test image sequence Claire is compared in Fig. 4. Hence, we propose a new difference quadtree encoding algorithm that exploits the previous frame's quadtree to generate the current frame's quadtree.

We can transmit only the difference quadtree, instead of the quadtree itself, to the decoder and the decoder can use it to generate the current quadtree from the previous frame's quadtree. Fig. 5 shows the difference quadtree between two successive quadtrees. The subquadtree rooted at node A in the previous frame's quadtree Q_{n-1} will be preserved in the current frame's quadtree Q_n. Node A is called a frozen node in the difference quadtree. The leaf node B will be connected with a new subquadtree in Q_n. Node B is called a live node. Although node C in Fig. 5(c) is a stationary leaf node, we can assume that this node C has four stationary children when finding the difference quadtree. Hence, the difference subquadtree for node C is shown as Fig. 5(b).

In [26], the splitting and shrinking operations are used to vary a binary tree structure. The same operations can also be applied to the quadtree structure. If a node in the difference quadtree is a live node, then this node will be connected with a new subquadtree. This operation can be called a splitting operation. The purpose of the splitting operation is to generate

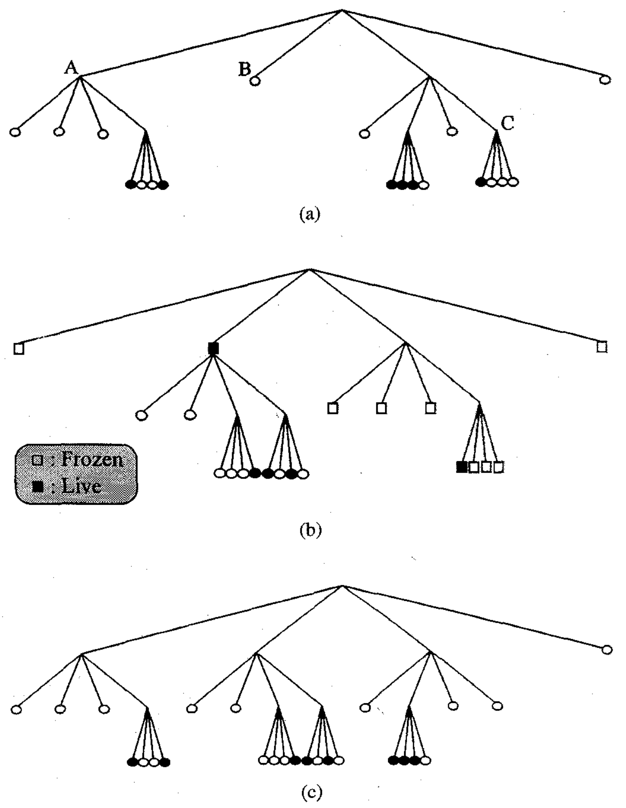


Fig. 5. (a) A quadtree structure for an image frame $n-1$. The subtree A will be preserved and node B will be connected with a new subquadtree at the next frame. The subquadtree of node C will be replaced with an updated subquadtree at frame n. (b) The difference quadtree that is transmitted to the decoder. (c) The regenerated quadtree for the current frame n in the decoder.

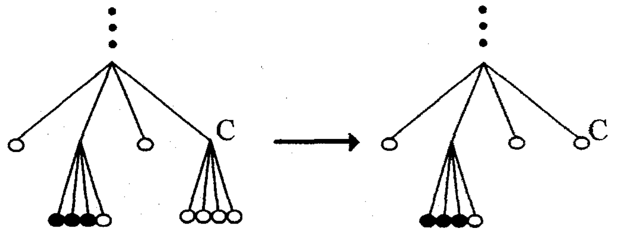


Fig. 6. Shrinking operation.

a new subquadtree. That is, if a node in the difference quadtree is live, then a new subquadtree will append to this node. The shrinking operation is also needed for pruning superfluous subquadtrees in Q_n. If all the child leaves of a node k are stationary, then all the children should be deleted and node k will be a stationary leaf. This shrinking operation should be repeated until all nodes do not have four stationary child leaves. For example, from Fig. 5(a) and (b), we can get a subquadtree with four stationary children. The subquadtree should be merged to a stationary node as shown in Fig. 6.

Now, we explain how to obtain the difference quadtree from the previous frame's quadtree Q_{n-1} and the current frame's Q_n. In the following algorithm, if a node is marked, then this node will be examined to decide its node type.

1) Step 1: Marked the root of the previous frame's quadtree Q_{n-1}.
2) Step 2: Select a marked node k and examine it. Let the subquadtree rooted at the node k be SQ_k.
3) Step 3: If the subquadtree SQ_k in Q_{n-1} is the same as the subquadtree SQ_k in Q_n, then the node k is a frozen node and we go to Step 6. Note that if the SQ_k in Q_n does not exist, then we can assume that this SQ_k in Q_n is a subquadtree with all children being stationary.
4) Step 4: If the node k in Q_{n-1} is a leaf node and the subquadtree SQ_k in Q_{n-1} is different from the subquadtree SQ_k in Q_n, then the node k is a live node and we go to Step 6.
5) Step 5: The four child nodes of the node k in Q_{n-1} need to be further examined and marked.
6) Step 6: Unmark the node k. If there are any marked nodes, we go to Step 2. Otherwise, halt.

For example, when the above algorithm checks node A at Step 3, node A will be set to a frozen node because the subquadtrees in frames $n-1$ and n are the same. When the algorithm checks the node B at Step 4, the node B will be set to a live node because it is a leaf in frame $n-1$ and there is a new subquadtree in frame n. For node C, we can assume that this node has four stationary children in frame n because the node C is a leaf node in frame n. When these four children are compared with the subquadtree of node C in frame $n-1$ will generate the difference subquadtree in Fig. 5(b).

The encoding algorithm of the interframe difference quadtree EBSMCVQ can be given as follows. For the first frame, EBSMCVQ is used to encode the first frame at a low bit rate and with high image quality. The quadtree for the edge information will be transmitted to the decoder. In general, the edge information is very similar to the moving information. Hence, the edge information for the first frame will be used as the moving information that is used by the next frame to generate its moving information.

For the other successive frames, the encoding algorithm of the interframe difference quadtree EBSMCVQ can be given as follows:

1) Step 1: Do a Sobel edge detection for the current frame and a preset threshold TH_e is used to decide whether the type of each block is edge or nonedge.
2) Step 2: For each edge block $\boldsymbol{b}_n$ in the current frame n, if $\boldsymbol{b}_{n-1}$ is a nonedge block or the distortion of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$ is larger than TH_{me}, where $\boldsymbol{b}_{n-1}$ is the block at same position as $\boldsymbol{b}_n$ in the frame $n-1$, then $\boldsymbol{b}_n$ is a moving block. Otherwise, $\boldsymbol{b}_n$ is a stationary block.
3) Step 3: For each nonedge block $\boldsymbol{b}_n$ in the current frame n, if $\boldsymbol{b}_{n-1}$ is an edge block or the distortion of $\boldsymbol{b}_n$ and $\boldsymbol{b}_{n-1}$ is larger than TH_{mn}, then $\boldsymbol{b}_n$ is a moving block. Otherwise, $\boldsymbol{b}_n$ is a stationary block.
4) Step 4: Generate the quadtree Q_n for the moving and edge information of the current frame. Compare Q_n with the previous frame's quadtree Q_{n-1}. Transmit the difference quadtree to the decoder.
5) Step 5: Use EBSMCVQ to encode all the moving blocks. For each moving block $\boldsymbol{y}$, if it is a nonedge block, then

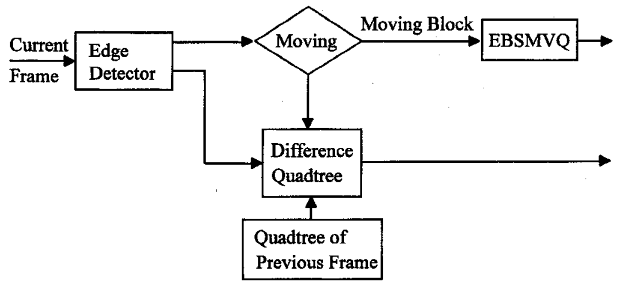


Fig. 7. The encoder of the interframe difference quadtree EBSMCVQ.

the nonedge master codebook C is used. If moving block $\boldsymbol{y}$ is an edge block, then its neighboring blocks' edge types are used to find the class number k (0–15) and master codebook C_k. Use the side-match operation to select the codewords of the state codebook SC from the master codebook C or C_k. Find the codeword $\hat{\boldsymbol{y}}$ or the current moving block $\boldsymbol{y}$.

6) Step 6: If there is another frame in the image sequence, we go to Step 1.

The interframe difference quadtree EBSMCVQ is shown in Figs. 7 and 8. Next, we describe the decoding algorithm as follows:

1) Step 1: Receive the difference quadtree and use it and the previous frame's quadtree Q_{n-1} to generate the current frame's quadtree Q_n. And then generate the moving and edge information for the current frame.
2) Step 2: Use EBSMCVQ to decode all the moving blocks. For each moving block $\boldsymbol{y}$, if it is a nonedge block, then the nonedge master codebook C is used. If moving block $\boldsymbol{y}$ is an edge block, then its neighboring blocks' edge types are used to find the class number k (0–15) and master codebook C_k. Use the side-match operation to select the codewords of the state codebook SC from the master codebook C or C_k. Use the index p sent by the encoder to find the codeword $\hat{\boldsymbol{y}}$ in the state codebook SC for the current moving block $\boldsymbol{y}$.
3) Step 3: If there is another frame to be decoded, we go to Step 1.

IV. Simulation Results

By computer simulations, the proposed interframe difference quadtree EBSMCVQ has been applied to two 176 × 144 monochrome image sequences, Claire and Miss America, with 256 gray levels. The training set for the codebook design algorithm consists of 45 monochrome image frames of size 176 × 144. Note that the blocks are 4 × 4 pixels. The image sequence Claire is in the training set but the image sequence Miss America is not in the training set.

Here the proposed interframe difference quadtree EBSMCVQ coding scheme with master codebook size 256 will be compared with two similar image sequence coding schemes using classified VQ and ordinary VQ. In the interframe classified VQ, all the moving blocks are encoded by the classified VQ with codebook size 16, instead of EBSMCVQ. In the interframe VQ, all the moving blocks are encoded by ordinary

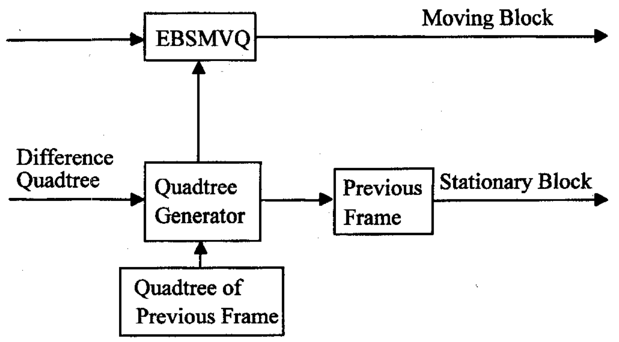


Fig. 8. The decoder of the interframe difference quadtree EBSMCVQ.

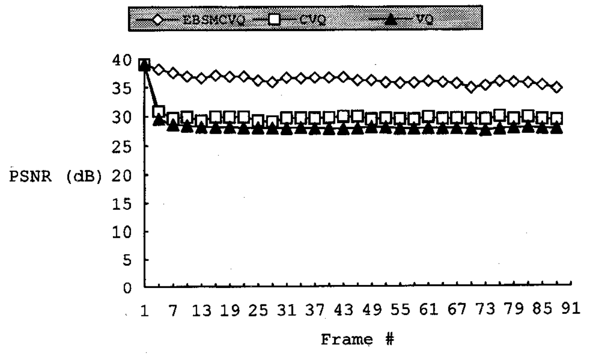


Fig. 9. The PSNR of three algorithms with $TH_{me} = 800$ and $TH_{mn} = 800$ for the luminance component of image sequence Claire as a function of the frame number.

VQ with codebook size 16. The size of the codebook in the VQ and classified VQ is very small in order to compare the these three methods at approximately the same bit rate. So as to improve the image quality of the first frame, it is encoded by the EBSMCVQ for these three methods. The peak signal-to-noise ratio (PSNR) of a reconstructed image with respect to its original image is used to evaluate the performance. The PSNR is defined as

$$\text{PSNR} = 10\log_{10}\frac{255^2}{\text{MSE}}\,\text{dB}. \tag{5}$$

Note that mean-square error for an $m \times n$ image is defined as

$$\text{MSE} = \left(\frac{1}{m \times n}\right)\sum_{i=1}^{m}\sum_{j=1}^{n}(x_{ij} - \hat{x}_{ij})^2 \tag{6}$$

where x_{ij} and $\hat{x}_{ij}$ denote the original and quantized gray levels, respectively.

The frame rate of the original test image sequences is 30 frames per second. Only every third frame is encoded and the other frames are skipped. That is, the temporal sampling rate is 1/3. The simulation results of the two testing image sequences with the EBSMCVQ, CVQ, and VQ are summarized in Tables I–IV. The reason for poor results of CVQ and VQ is the very small size of the codebooks for comparing these three methods at the very low bit rate. Tables I and II and Tables III and IV, respectively, show the results for different threshold values that are used to decide whether a block is moving or

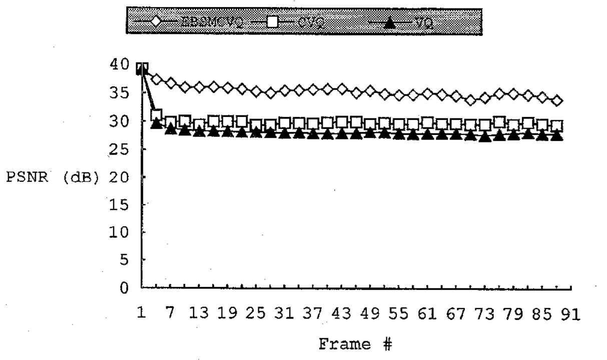


Fig. 10. The PSNR of three algorithms with $TH_{me} = 800$ and $TH_{mn} = 3\,000$ for the luminance component of image sequence Claire as a function of the frame number.

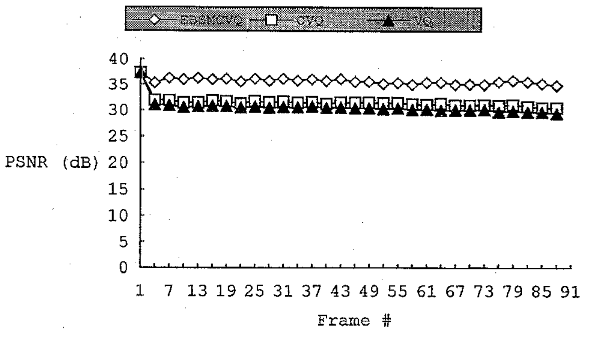


Fig. 11. The PSNR of three algorithms with $TH_{me} = 800$ and $TH_{mn} = 800$ for the luminance component of image sequence Miss America as a function of the frame number.

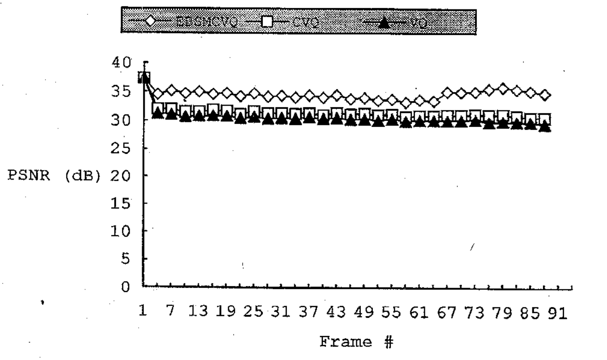


Fig. 12. The PSNR of three algorithms with $TH_{me} = 800$ and $TH_{mn} = 3\,000$ for the luminance component of image sequence Miss America as a function of the frame number.

TABLE I
THREE ALGORITHMS ARE COMPARED WITH $TH_{me} = 800$ AND $TH_{mn} = 800$ FOR THE LUMINANCE COMPONENT OF IMAGE SEQUENCE CLAIRE

	EBSMCVQ	CVQ	VQ
Average number of bits for moving blocks per frame	959.3	1218.4	1309.6
Average number of bits for difference quadtree per frame	283.7	213.9	205.1
Bit rate (bit/pixel)	0.0490	0.0565	0.0598
PSNR (dB)	36.33	29.96	28.41

TABLE II
THREE ALGORITHMS ARE COMPARED WITH $TH_{me} = 800$ AND $TH_{mn} = 3\,000$ FOR THE LUMINANCE COMPONENT OF IMAGE SEQUENCE CLAIRE

	EBSMCVQ	CVQ	VQ
Average number of bits for moving blocks per frame	739.5	912.4	963.0
Average number of bits for difference quadtree per frame	257.6	223.7	211.2
Bit rate (bit/pixel)	0.0393	0.0448	0.0463
PSNR (dB)	35.40	30.08	28.48

TABLE III
THREE ALGORITHMS ARE COMPARED WITH $TH_{me} = 800$ AND $TH_{mn} = 800$ FOR THE LUMINANCE COMPONENT OF IMAGE SEQUENCE MISS AMERICA

	EBSMCVQ	CVQ	VQ
Average number of bits for moving blocks per frame	1607.8	1593.1	1688.9
Average number of bits for difference quadtree per frame	453.7	411.7	405.1
Bit rate (bit/pixel)	0.0813	0.0791	0.0826
PSNR (dB)	35.70	31.57	30.62

TABLE IV
THREE ALGORITHMS ARE COMPARED WITH $TH_{me} = 800$ AND $TH_{mn} = 3\,000$ FOR THE LUMINANCE COMPONENT OF IMAGE SEQUENCE MISS AMERICA

	EBSMCVQ	CVQ	VQ
Average number of bits for moving blocks per frame	1333.6	1274.3	1307.6
Average number of bits for difference quadtree per frame	383.5	373.1	374.1
Bit rate (bit/pixel)	0.0678	0.0650	0.0664
PSNR (dB)	34.61	31.43	30.58

stationary. These thresholds TH_{me} and TH_{mn} can be used to adjust the bit rate. Because edge information is more important than nonedge information, a larger threshold TH_{mn} value for nonedge blocks is used to reduce the bit rate. In Tables II and IV, we can see that the bit rate is much reduced while acceptable image quality is still obtained. Figs. 9–12 show the PSNR values for these algorithms at approximately the same bit rate.

The comparisons of the difference quadtree technique and the complete quadtree technique are described in Tables V and VI. In the complete quadtree technique, an independent quadtree for the moving and edge information of each frame is needed and the correlations of successive quadtrees is ignored. The number of bits for the difference quadtree is only 25–50% of that for the complete quadtree. The bit rate for the difference quadtree is only 0.0112–0.0179 b/pixel. Hence, the difference quadtree technique can effectively reduce the overhead for the moving and edge information.

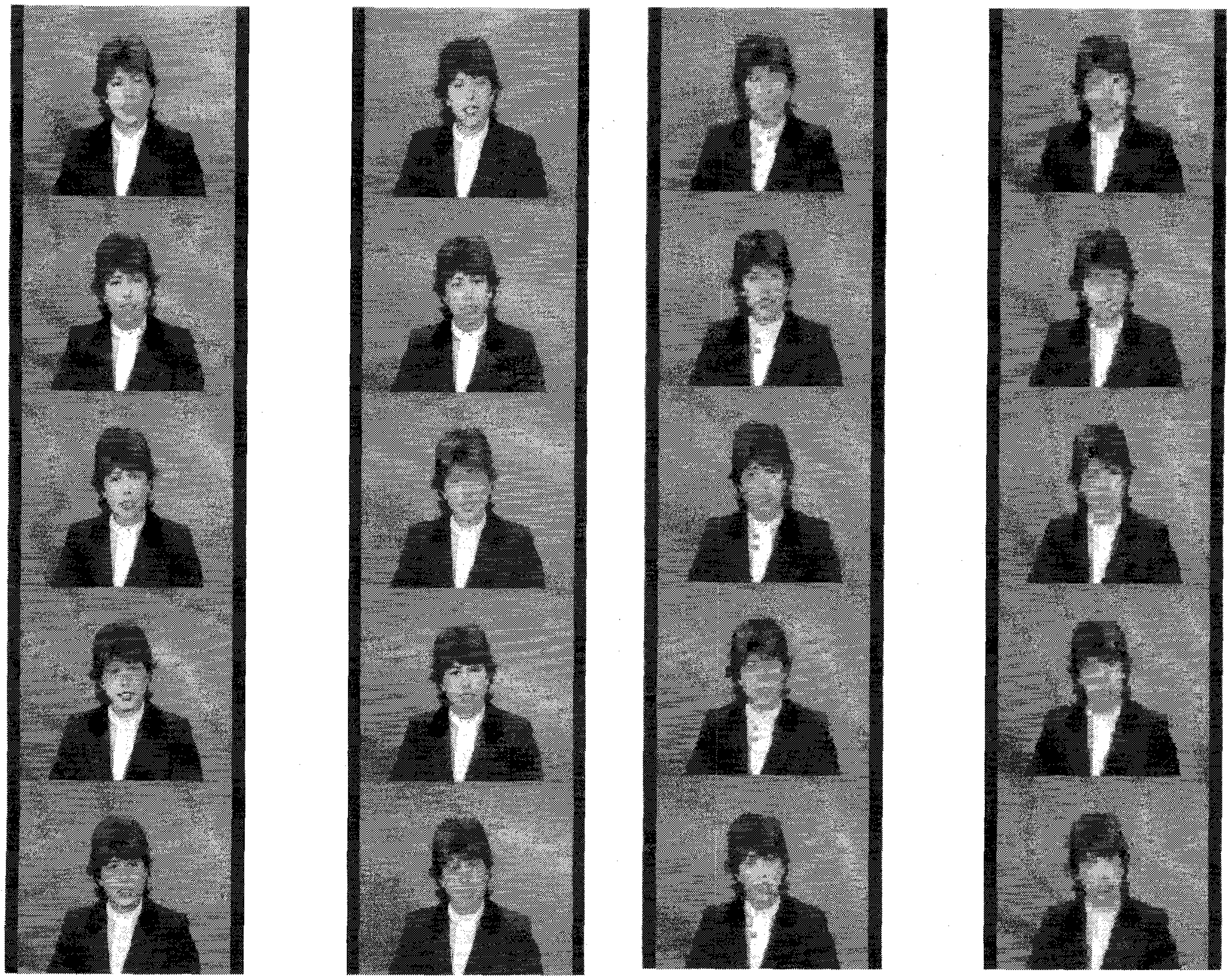


Fig. 13. Five original and encoded frames 28–40 of the image sequence Claire. (a) Original frames. (b) Encoded frames using EBSMCVQ with master codebook size 256 at 0.0490 b/pixel. (c) Encoded frames using CVQ with codebook size 16 at 0.0565 b/pixel. (d) Encoded frames using VQ with codebook size 16 at 0.0598 b/pixel.

TABLE V
COMPARISON FOR THE NUMBER OF BITS OF THE DIFFERENCE QUADTREE AND COMPLETE QUADTREE PER FRAME TO REPRESENT THE MOVING AND EDGE INFORMATION FOR THE LUMINANCE COMPONENT OF IMAGE SEQUENCE CLAIRE

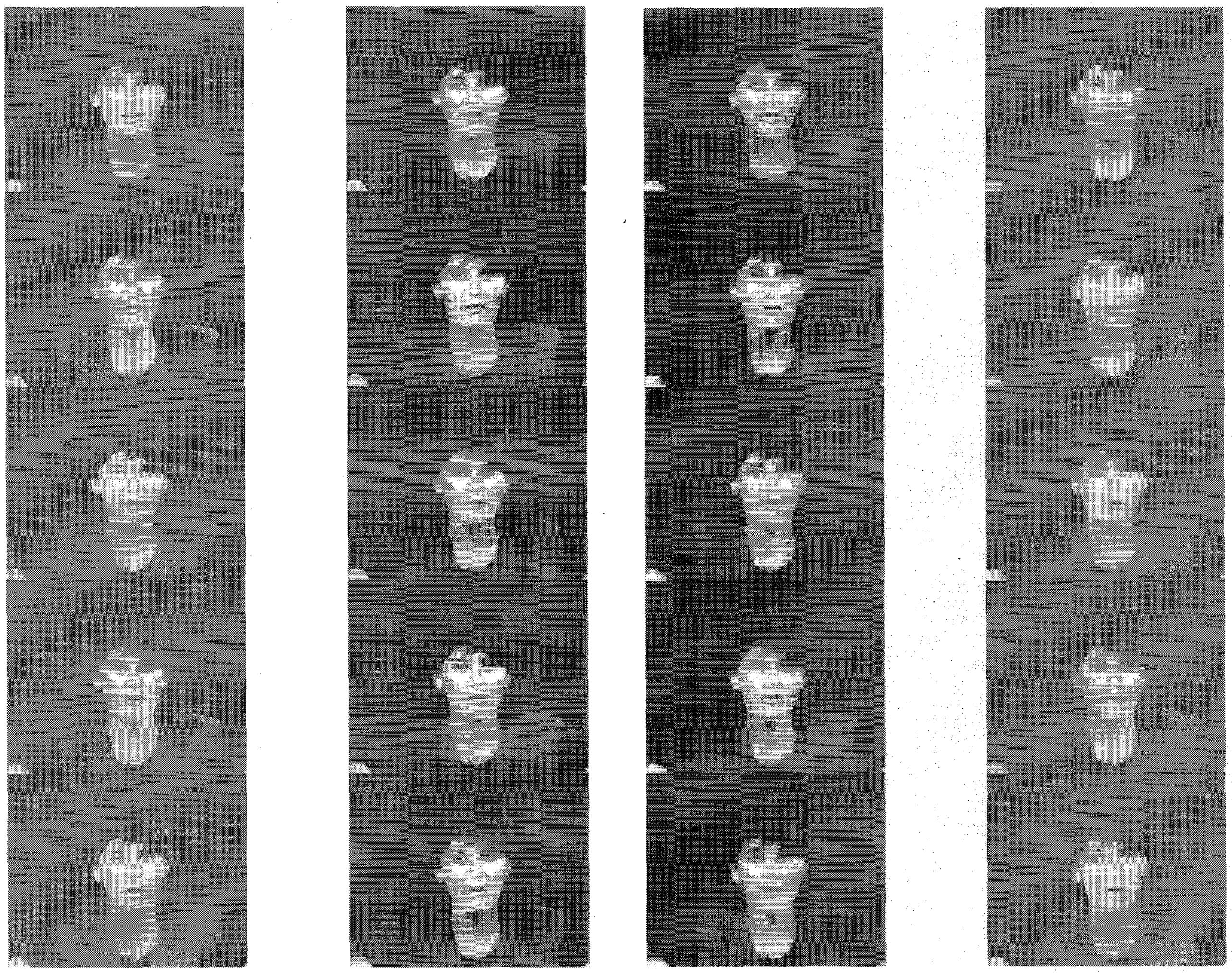
	TH_{me} = 800 and TH_{mn} = 800			TH_{me} = 800 and TH_{mn} = 3000		
	EBSMCVQ	CVQ	VQ	EBSMCVQ	CVQ	VQ
Difference quadtree	283.7	213.9	153.9	257.6	223.7	211.2
Complete quadtree	676.0	843.1	870.1	558.1	704.3	707.5

Fig. 13(a) shows the original frames 28–40 of the image sequence Claire and Fig. 13(b) and (c) shows the encoded frames using EBSMCVQ at 0.0490 b/pixel, CVQ at 0.0565 b/pixel, and VQ at 0.0598 b/pixel, respectively. Fig. 14 shows the original frames and the encoded frames of image sequence Miss America using EBSMCVQ at 0.0813 b/pixel, CVQ at 0.0791 b/pixel, and VQ at 0.0826 b/pixel, respectively.

TABLE VI
COMPARISON FOR THE NUMBER OF BITS OF THE DIFFERENCE QUADTREE AND COMPLETE QUADTREE PER FRAME TO REPRESENT THE MOVING AND EDGE INFORMATION FOR THE LUMINANCE COMPONENT OF IMAGE SEQUENCE MISS AMERICA

	TH_{me} = 800 and TH_{mn} = 800			TH_{me} = 800 and TH_{mn} = 3000		
	EBSMCVQ	CVQ	VQ	EBSMCVQ	CVQ	VQ
Difference quadtree	453.7	411.7	405.1	383.5	373.1	374.1
Complete quadtree	829.1	989.6	1048.5	713.6	801.6	840.8

The interframe EBSMCVQ is also compared with an implementation of the H.261 standard, which is from the Portable Video Research Group [27], at approximately the same PSNR for the same test image sequences in Table VII. In H.261, we count only the luminance component Y for the bit rate



(a) (b) (c) (d)

Fig. 14. Five original and encoded frames 28–40 of the image sequence Miss America. (a) Original frames. (b) Encoded frames using EBSMCVQ with master codebook size 256 at 0.0813 b/pixel. (c) Encoded frames using CVQ with codebook size 16 at 0.0791 b/pixel. (d) Encoded frames using VQ with codebook size 16 at 0.0826 b/pixel.

TABLE VII
COMPARISON OF PSNR AND BIT RATE FOR EBSMCVQ AND H.261

Image Sequence		EBSMCVQ	H.261
Claire	PSNR (dB)	35.40	34.99
	Bit rate (bit/pixel)	0.0393	0.1474
Miss America	PSNR (dB)	34.61	34.91
	Bit rate (bit/pixel)	0.0678	0.1340

and PSNR. We find that the bit rate of our method is much smaller than that of H.261.

V. CONCLUSIONS

In this paper, we apply the EBSMCVQ to image sequence coding. Each block is classified into a moving block or stationary block. Only the moving blocks will be encoded by EBSMCVQ so as to reduce the bit rate. The moving and edge information for each frame will be encoded by a quadtree data structure. The information for the quadtree can be further reduced by using the previous frame's quadtree. Hence, we propose a new difference quadtree technique that uses the previous quadtree to generate the current quadtree. Moreover, the interframe difference quadtree EBSMCVQ is very suitable for very low bit rate image sequence encoding. In the experiment, the bit rate is only 0.0393 b/pixel for the image sequence Claire and the PSNR is still up to 35.40 dB.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers whose comments led to a greatly improved manuscript.

References

[1] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1980.

[2] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, pp. 4–29, Apr. 1984.

[3] M. Goldberg, P. R. Boucher, and S. Shlien, "Image compression using adaptive vector quantization," *IEEE Trans. Commun.*, vol. COM-34, no. 2. pp. 180–187, Feb. 1986.

[4] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: A review," *IEEE Trans. Commun.*, vol. 36, no. 8, pp. 957–971, Aug. 1988.

[5] H. M. Hang and B. G. Haskell, "Interpolative vector quantization of color images," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 465–470, Apr. 1988.

[6] D. Miller and K. Rose, "Joint source-channel vector quantization using deterministic annealing," in *Proc. ICASSP*, vol. III, 1992, pp. 377–380.

[7] R. A. Vander Kam, P. A. Chou, E. A. Riskin, and R. M. Gray, "An algorithm for joint vector quantizer and halftoner design," in *Proc. ICASSP*, vol. III, 1993, pp. 497–500.

[8] K. L. Oehler and R. M. Gray, "Mean-gain-shape vector quantization," in *Proc. ICASSP*, vol. V, 1993, pp. 241–244.

[9] C. Lo, "Image coding with a map and vector quantizer," in *Proc. ICASSP*, vol. V, 1993, pp. 606–608.

[10] M. O. Dunham and R. M. Gray, "An algorithm for the design of label-transition finite-state vector quantizers," *IEEE Trans. Commun.*, vol. COM-33, no. 1, pp. 83–89, Jan. 1985.

[11] J. Foster, R. M. Gray, and M. O. Dunham, "Finite-state vector quantization for waveform coding," *IEEE Trans. Inform. Theory*, vol. IT-31, no. 3, pp. 348–359, May 1985.

[12] R. Aravind and A. Gersho, "Low-rate image coding with finite-state vector quantization," in *Proc. ICASSP*, 1986, pp. 137–140.

[13] ——, "Image compression based on vector quantization with finite memory," *Opt. Eng.*, vol. 26, no. 7, pp. 570–580, July 1987.

[14] Y. Hussain and N. Farvardin, "Variable-rate finite-state vector quantization of images," in *Proc. ICASSP*, 1991, pp. 2301–2304.

[15] T. Kim, "Side match and overlap match vector quantizers for images," *IEEE Trans. Image Processing*, vol. 1, no. 2, pp. 170–185, Apr. 1992.

[16] R. F. Chang and W. M. Chen, "Adaptive quadtree-based side-match finite-state vector quantization," in *SPIE's Symp. Visual Commun. Image Processing*, 1994, pp. 377–388.

[17] B. Ramamurthi and A. Gersho, "Classified vector quantization of images," *IEEE Trans. Commun.*, vol. COM-34, no. 11, pp. 1105–1115, Nov. 1986.

[18] H. F. Sun and M. Goldberg, "Adaptive vector quantization for image sequence coding," in *Proc. ICASSP*, Mar. 1985, pp. 133–136.

[19] M. Goldberg and H. F. Sun, "Image sequence coding using vector quantization," *IEEE Trans. Commun.*, vol. COM-34, pp. 703–710, July 1986.

[20] P. Strobach, "Tree-structured scene adaptive coder," *IEEE Trans. Commun.*, vol. 38, no. 4, pp. 477–486, Apr. 1990.

[21] N. M. Nasrabadi, C. Y. Choo, and J. U. Roy, "Interframe hierarchical address-vector quantization," *IEEE J. Select. Areas Commun.*, vol. 10, no. 5, pp. 960–967, June 1992.

[22] S. Liu and M. Hayes, "Segmentation-based coding of motion difference and motion field images for low-rate video compression," in *Proc. ICASSP*, vol. III, 1992, pp. 525–528.

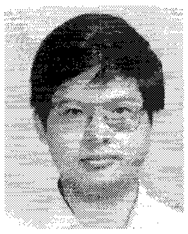
[23] X. Zhang, M. C. Cavenor, and J. F. Arnold, "Adaptive quadtree coding of motion-compensated image sequences for use on the broadband ISDN," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 3, pp. 222–229, June 1993.

[24] F. G. B. De Natale, G. S. Desoli, S. Fioravanti, and D. D. Giusto, "An edge-based splitting criterion for adaptive transform coding," in *Proc. ICASSP*, vol. V, 1993, pp. 409–412.

[25] S. Liu and M. Hayes, "Video compression using quadtree segmentation and component quantization," in *Proc. ICASSP*, vol. V, 1993, pp. 429–432.

[26] R. F. Chang, W. T. Chen, and J. S. Wang, "Image sequence coding using adaptive nonuniform tree-structured vector quantization," *J. Visual Commun. Image Representation*, pp. 166–176, June 1991.

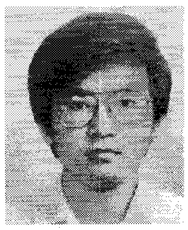
[27] A. C. Hung, "PVRG-P64 Codec 1.1," Portable Video Res. Group, Mar. 1, 1993.



Ruey-Feng Chang (S'90–M'91) was born in Taichung, Taiwan, on August 25, 1962. He received the B.S. degree in electrical engineering from National Cheng Kung University, Taiwan, Republic of China, in 1984, the M.S. degree in computer and decision sciences and the Ph.D. degree in computer science from National Tsing Hua University, Taiwan, Republic of China, in 1988 and 1992, respectively.

He is currently an Associate Professor of the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan, Republic of China. His research interests include still image coding, video sequence coding, and packet video.

Dr. Chang is a member of SPIE and Phi Tau Phi.



Wei-Ming Chen (S'91–M'95) was born in Taipei, Taiwan, on August 8, 1965. He received the B.S. degree in electrical engineering from National Taiwan Institute of Technology, Taiwan, Republic of China, in 1992, and the M.S. degree in computer science and information engineering from National Chung Cheng University, Taiwan, Republic of China, in 1994.

Currently, he is a member of technical staff in the computer center at National Dong Hwa University, Taiwan, Republic of China. His research interest includes image sequence coding.