View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by NASA Technical Reports Server

(NASA-TM-111108) MODEL-BASED VQ FOR IMAGE DATA ARCHIVAL, RETRIEVAL AND DISTRIBUTION (Bowie State Univ.) 10 p

N96-11488

Unclas

G3/82 0068424

ORIGINAL PAGE IS OF POOR QUALITY

NASA-TM-111108

IN-82-TM
5080
P. 10

Model-based VQ for image data archival, retrieval and distribution

Mareboyana Manohar
Bowie State University
Department of Computer Science
Bowie, MD 20715

James C. Tilton
NASA Goddard Space Flight Center
Information Sciences and Technology Branch
Greenbelt, MD 20771

ABSTRACT

An ideal image compression technique for image data archival, retrieval and distribution would be one with the asymmetrical computational requirements of Vector Quantization (VQ), but without the complications arising from VQ codebooks. Codebook generation and maintenance are stumbling blocks which have limited the use of VQ as a practical image compression algorithm. Model-based VQ (MVQ), a variant of VQ described here, has the computational properties of VQ but does not require explicit codebooks. The codebooks are internally generated using mean removed error and Human Visual System (HVS) models. The error model assumed is the Laplacian distribution with mean, λ, computed from a sample of the input image. A Laplacian distribution with mean, λ, is generated with a uniform random number generator. These random numbers are grouped into vectors. These vectors are further conditioned to make them perceptually meaningful by filtering the DCT coefficients from each vector. The DCT coefficients are filtered by multiplying by a weight matrix that is found to be optimal for human perception. The inverse DCT is performed to produced the conditioned vectors for the codebook. The only image dependent parameter used in the generation of codebook is the mean, λ, that is included in the coded file to repeat the codebook generation process for decoding.

Keywords: Image data compression, vector quantization, model-based vector quantization, image data archival and retrieval.

1. INTRODUCTION

We have found Vector Quantization (VQ) to be an attractive lossy compression technique for image data archival, retrieval and distribution[1] due to its asymmetric computational property. While VQ compression can be computationally demanding, decompression of the VQ coded data is a computationally light table lookup process. This means less computational burden on the user compared to some compression techniques that are characterized by symmetric computational requirements such as JPEG/DCT[2]. Thus VQ is ideal for archive and distribution, where images are compressed once at the archiving center and potentially decompressed many times at different remote sites.

The performance of VQ depends heavily on the quality of codebooks employed. The codebook generation involves a computationally intensive training process. One of the popular training algorithms for VQ codebook generation is the Generalized Lloyd Algorithm (GLA)[3]. See[4,5] for other training algorithms. The training process is carried out on a set of image data called the training data set. The codebook thus generated provides optimal performance (in the rate distortion sense) for a given size of the codebook for all the images included in the training set. However, for images outside this training set the performance is suboptimal. By carefully selecting the training set, however, an acceptable level of performance can be achieved.

Codebook size is an important issue in optimal performance of VQ. The compression ratio (CR, defined as the ratio of number of bits in the input image to the number of bits in the compressed image) for VQ techniques decreases as the logarithm of the codebook size and distortion (normally measured as mean squared error between the input image and reconstructed image) decreases linearly with increasing

ORIGINAL PAGE IS
OF POOR QUALITY

0-8194-1841-2/95/$6.00

codebook size[6]. Therefore, it is important to consider large codebooks for better rate-distortion performance of the VQ. The large codebooks, however, require large computational requirements for coding. We have used a parallel full search implementation of VQ to solve the computational problems arising from large codebooks[7]. Further, since the codebooks involved are large, we cannot afford to include codebook in every image that we compress to decode. Thus, generation and maintenance of codebooks pose some significant problems for VQ to be a viable option for data archival, retrieval and distribution.

VQ would be a more attractive approach for image data archival, retrieval and distribution if it did not require the explicit generation and maintenance of codebooks. We are thus faced with the question of whether we can develop a variant of VQ that does not explicitly depend on the use of codebooks. Our work in this direction has resulted in what we call Model-Based VQ (MVQ). MVQ is based on generating mean removed residual vectors using error models such as Laplacian. The error values are generated using uniform random number generator and shaped according to Laplacian distribution by using an inverse function for filtering. The generated error values are then normalized to -1.0 to 1.0. The error values thus obtained are independent and identically distributed (i.i.d.). Since we are trying to model actual error residuals from the source which are not independent, we need to process these error values to impose the actual error characteristics. In our previous work[8] on MVQ we computed the covariance matrix of the mean removed vectors from the source and factorized it into lower and upper triangular matrices. The lower triangular matrix was used to impose correlations among the i.i.d. Laplacian numbers. This model was called Laplacian Multivariate Model (LMV). The vectors generated from this model constituted a codebook, which was used to VQ encode the source image. This codebook can be completely specified by the seed point used for generating the uniform random numbers from -1.0 to 1.0, λ (the mean of the Laplacian distribution), and finally the lower triangular matrix (L) . These parameters are included in the coded file of the source image and so the decoder can generate the codebook that is used for coding at the decoding side from the model parameters and using the block means along with indexes to the codebook entries that matched the vectors derived from the source image. The results obtained using this model were reasonable, but were not as good as VQ for some NASA images (Thematic Mapper data).

We propose here that the model can be improved further by perceptually weighting the residual vectors from the model. In the remaining sections we describe MVQ that uses the perceptual weighting of the model generated residual vectors and provide results on set of NASA Earth science images. We also compare these results with VQ results on the same image data set.

2. MODEL-BASED VQ ALGORITHM

The MVQ algorithm implicitly generates its codebook using mathematical models based on a few image specific parameters from the source which are transmitted along with the coded image data so that decoding can be performed by generating the same codebook at decoding end. The model simulates the mean removed vectors of the source. The means (block means) from the source, m_i of vector $\mathbf{X}_i$, are scalar random variables given by

$$m_i = {}^1/_k \sum_j x_{ij} \qquad (1)$$

where x_{ij} is j^{th} element of the vector $\mathbf{X}_i$. For all the vectors, we compute the vector means and transmit them after lossless compression such as JPEG/DPCM or Ziv-Lempel algorithm[10]. The second phase of the algorithm is to send residual information in compressed form. The residual vector for i^{th} block composed of scalar random variables is given by

ORIGINAL PAGE IS
OF POOR QUALITY

codebook size[6]. Therefore, it is important to consider large codebooks for better rate-distortion performance of the VQ. The large codebooks, however, require large computational requirements for coding. We have used a parallel full search implementation of VQ to solve the computational problems arising from large codebooks[7]. Further, since the codebooks involved are large, we cannot afford to include codebook in every image that we compress to decode. Thus, generation and maintenance of codebooks pose some significant problems for VQ to be a viable option for data archival, retrieval and distribution.

VQ would be a more attractive approach for image data archival, retrieval and distribution if it did not require the explicit generation and maintenance of codebooks. We are thus faced with the question of whether we can develop a variant of VQ that does not explicitly depend on the use of codebooks. Our work in this direction has resulted in what we call Model-Based VQ (MVQ). MVQ is based on generating mean removed residual vectors using error models such as Laplacian. The error values are generated using uniform random number generator and shaped according to Laplacian distribution by using an inverse function for filtering. The generated error values are then normalized to -1.0 to 1.0. The error values thus obtained are independent and identically distributed (i.i.d.). Since we are trying to model actual error residuals from the source which are not independent, we need to process these error values to impose the actual error characteristics. In our previous work[8] on MVQ we computed the covariance matrix of the mean removed vectors from the source and factorized it into lower and upper triangular matrices. The lower triangular matrix was used to impose correlations among the i.i.d. Laplacian numbers. This model was called Laplacian Multivariate Model (LMV). The vectors generated from this model constituted a codebook, which was used to VQ encode the source image. This codebook can be completely specified by the seed point used for generating the uniform random numbers from -1.0 to 1.0, λ (the mean of the Laplacian distribution), and finally the lower triangular matrix (L) . These parameters are included in the coded file of the source image and so the decoder can generate the codebook that is used for coding at the decoding side from the model parameters and using the block means along with indexes to the codebook entries that matched the vectors derived from the source image. The results obtained using this model were reasonable, but were not as good as VQ for some NASA images (Thematic Mapper data).

We propose here that the model can be improved further by perceptually weighting the residual vectors from the model. In the remaining sections we describe MVQ that uses the perceptual weighting of the model generated residual vectors and provide results on set of NASA Earth science images. We also compare these results with VQ results on the same image data set.

2. MODEL-BASED VQ ALGORITHM

The MVQ algorithm implicitly generates its codebook using mathematical models based on a few image specific parameters from the source which are transmitted along with the coded image data so that decoding can be performed by generating the same codebook at decoding end. The model simulates the mean removed vectors of the source. The means (block means) from the source, m_i of vector $\mathbf{X}_i$, are scalar random variables given by

$$m_i = {}^1/_k \sum_j x_{ij} \qquad (1)$$

where x_{ij} is j^{th} element of the vector $\mathbf{X}_i$. For all the vectors, we compute the vector means and transmit them after lossless compression such as JPEG/DPCM or Ziv-Lempel algorithm[10]. The second phase of the algorithm is to send residual information in compressed form. The residual vector for i^{th} block composed of scalar random variables is given by

ORIGINAL PAGE IS
OF POOR QUALITY

$$\mathbf{R}_i = \mathbf{X}_i - m_i \mathbf{U} \tag{2}$$

where $\mathbf{U}$ is a unit vector of same dimension as input vector $\mathbf{X}_i$. In MVQ, the residual vector, $\mathbf{R}_i$, is represented by an index I to the model generated codebook (CB) entries:

$$\text{MVQ:}\ \ \mathbf{R}_i \dashrightarrow I \tag{3}$$

In our implementation, the codebook size (nc, number of entries in the codebook) is 16,384 resulting in 14 bit representation for the index, I. The compression ratio (CR) that can be obtained from this scheme is given by

$$CR = \frac{k}{b + \log_2(nc)} \tag{4}$$

At the decoding end,

$$\mathcal{R}_i = CB(I) \tag{5}$$

where, $\mathcal{R}_i$ is an approximation of $\mathbf{R}_i$. The reconstructed vector at the decoding end is given by

$$\mathcal{X}_i = m_i + \mathcal{R}_i \tag{6}$$

The distortion between input vector, $\mathbf{X}_i$ and the reconstructed vector, $\mathcal{X}_i$ is expressed in terms of mean squared error (MSE). The MSE for all the vectors drawn from the source image is given by

$$D = \sum_i \sum_j (x_{ij} - \hat{x}_{ij})^2 \tag{7}$$

The important element of the MVQ is the simulation of residual vectors of the source image and building the codebook of 16,384 codevectors from this process. We note why we settled for 16,384 vectors in the section (below) on implementation on the MasPar.

The input image is decomposed into square blocks of 4x4, 5x5, etc., depending on the targeted compression ratio. From Eq. 4, for a given compression ratio, one can compute the vector size, k (r x c), where r is the number of rows and c is the number of columns of the block of pixels from the source image data. For each block of pixels, the block mean is computed, compressed using appropriate lossless compression, and transmitted. The residual vectors are computed by removing the mean from the vector formed by raster scan of the image blocks. The residual vector elements are used to compute the mean assuming the residual vector elements fit into a Laplacian distribution,

$$p(r) = \frac{1}{2\lambda} e^{-|x|/\lambda} \tag{8}$$

where λ is the mean of the distribution. The mean is given by

$$\lambda = \frac{\sigma_e}{\sqrt{2}} \tag{9}$$

where σ_e is the standard deviation of the residual vector elements treated as i.i.d. random variables. After computing λ from the residual vectors of the image, it is used to generate Laplacian distribution using uniform random number generator. If u_i is the i^{th} uniform random number (value -1.0 to 1.0) generated by uniform random number generator, then the random number that forms is Laplacian distribution with mean equal to λ is given by

$$v_i = -\lambda \log_e (1-u_i) \tag{10}$$

A + or - sign is randomly given to v_i.

Number of v_i's generated depend on the codebook size and vector size. For a vector size of k and codebook size of nc, the number of v_i's is equal to the product of k and nc. The random numbers are grouped by k elements to form nc vectors that in turn form a codebook. These random numbers are independent and identically distributed, whereas the residual vector elements show considerable correlations among them which are manifest in nonzero off diagonal elements of covariance matrix. The most apparent solution to tuning the codebook to actual source is to impose correlations on the vector elements of the codebook so that they have approximately same covariance as the source residual vectors.

In our earlier work[8] we decomposed the covariance matrix of the mean removed residual vectors of the source into upper and lower triangular matrices and mapped all the codebook entries onto lower triangular matrix to impart the cross correlation structure of the input source to randomly generated numbers grouped into vectors. Let Σ_s be the covariance matrix of the source mean removed residual vectors (of size kxk). By using the Cholesky's decomposition, Σ_s is decomposed into lower (L) and upper (U) triangular matrices. The lower triangular matrix is used to condition the i.i.d. of the codebook vectors as follows.

$$C_i = \mathbf{L}\, C_i \tag{11}$$

Eq. 11 was applied to all the vectors of the codebook. The rate distortion results from this algorithm were reasonable, but not comparable to training VQ. The question that faced us was how to improve the codebook to get better performance. At this point we brought in the Human Visual Systems (HVS). Our conjecture was that the residual vectors generated should be perceptually meaningful for better performance.

2.1 The Role of HVS in the Model Codebook

The HVS system has been incorporated in JPEG/DCT with excellent results. In JPEG/DCT, the DCT coefficients in are quantized based on their significance to human perception. Human visual weighted DCT coefficients have been used for progressive image transmission by Chitprasert and Rao[9] with visually more pleasing results than unweighted DCT coefficients. In order to impose HVS properties on codebook vectors generated by model, the vectors have to be DCT transformed and weighted using the HVS DCT weight matrix and then inverse transformed. The HVS DCT weight matrix we have used for model generated residual vectors is shown in Fig. 1. Note that we have modified the coefficient weighting matrix in[9] by making the DC term 0 so that it can be applied to residual vectors with 0 DC value.

ORIGINAL PAGE IS
OF POOR QUALITY

0.000	1.000	0.702	0.381	0.186	0.085	0.037	0.016
1.000	0.455	0.308	0.171	0.084	0.039	0.017	0.007
0.702	0.308	0.212	0.124	0.064	0.031	0.014	0.063
0.381	0.171	0.124	0.077	0.042	0.021	0.010	0.004
0.185	0.084	0.064	0.042	0.025	0.013	0.007	0.003
0.084	0.039	0.031	0.021	0.013	0.007	0.004	0.002
0.037	0.017	0.014	0.010	0.006	0.004	0.002	0.001
0.016	0.007	0.006	0.004	0.003	0.002	0.001	0.0006

Figure 1: The Human Visual Systems weight function of DCT coefficients.

The model codebook using HVS is generated as follows. A set of (k*nc) uniformly distributed random numbers $\{u_i\}$ are generated and are nonlinearly mapped to $\{v_i\}$ according to Eq. 10 to fit into Laplacian distribution of specified mean λ which is computed from the input source as discussed. Next the $\{v_i\}$ are grouped to form set of nc k-element vectors $\{\mathbf{C}_i\}$. The vectors then undergo DCT transformation as given below.

$$\mathbf{C}_i^{dct} = \mathrm{DCT}(\mathbf{C}_i) \tag{12}$$

The next step is to weight each coefficient of and take the inverse DCT transform.

$$C_i = \mathrm{IDCT}(\mathbf{C}_i^{dct} * \mathrm{W}) \tag{13}$$

where * stands for element by element product of the two matrices. If $\mathbf{C}_i$ is smaller than 8x8 block (vector size less than 64), the corresponding subset of the W matrix are used in the product. The codebook containing $\{\mathbf{C}_i\}$ are now replaced by $\{C_i\}$ and the resulting codebook produces much better rate-distortion performance.

3. IMPLEMENTATION ON THE MASPAR

MVQ is an asymmetrical algorithm like VQ. While decoding is a table look-up process that can be performed quite efficiently on a sequential machine, coding using full search is computationally very intensive. Others solve VQ coding computational problems by structuring the codebook as tree[6]. However, the price of such structuring is suboptimal performance. We instead solve the computational problem by using an implementation on a massively parallel computer to obtain optimal performance (in the rate-distortion sense) for a given codebook size by doing full search on the codebook for best match of the source residual.

We have implemented MVQ coding on Goddard Space Flight Center's 16,384 processor MasPar MP-2. Each processing element (PE) has a local memory of 64 Kbytes. We generate 16,384 vectors of using the algorithm given in the preceding section and load each vector in each of the 16,384 PEs. We chose 16,384 vectors because we have found that this size codebook gives good performance, and because this size is most efficient on a 16,384 PE MasPar. The input image is decomposed into blocks of 4x4 or 6x6 or 8x8 pixels depending on the rate-distortion requirements and the block means are computed on the MasPar by propagating each vector from the image to one PE and the mean removed residuals from the input image are computed in parallel. The mean of the vectors are stored in a separate file and compressed using a JPEG's lossless compression[2] technique. The input residual vector elements are normalized with respect to maximum residual vector element value from the mean removed source. This can be accomplished by using MasPar's min, max functions that take time proportional to the product of number of bits per pixel and clock cycle of the MasPar. Now, the normalized residual vectors are coded one by one by propagating each one to all PEs. The distance

between the normalized residual vector from the input source and the codebook entries can be computed and the min distance can be found by using min function of the Maspar. The time taken for the best match search from the codebook takes time proportional to the number of bits in the Euclidean distance norm between source residual vector and codebook entry (32 bits).

If the codebook size is smaller than the PE array size by a factor of 2 or its multiple, simple load balancing techniques can be used to gain speedups for smaller codebooks. For example, if the codebook size is half the array size, then each codebook entry can be split up into two smaller codebook entries each containing half the number of elements and loaded into local memories of the adjacent PEs. The input residual vector from the source is similarly split into two smaller vectors and propagated such that first half of the vector is placed in PEs with even address and second half in those with odd address PEs. The distances are then computed in each PE separately and combined by shift operations to find the closest matching vector. Codebooks larger than PE array size by factor of 2 or its multiples can be handled by using processor virtualization.

In decoding, we retrieve the λ from coded file and generate the codebook using the λ and the DCT coefficient weighting function given in Fig. 1. After the codebook is generated from λ, the rest of the process of decoding is a simple table lookup process. From the coded file, index for each residual vector is retrieved and the corresponding residual vector is read from the codebook. When the block mean is added to this residual vector, the block is reconstructed.

4. RESULTS AND DISCUSSION

The MVQ algorithm was tested on some Landsat images. Results are shown for Washington, DC downtown image of Thematic Mapper (TM) image of band 4. The original image is shown in Fig. 2a. The compressed and reconstructed images are shown in Fig. 2b, c, and d for different compression ratios (or different vector sizes). The results are comparable to the more standard VQ in which the codebook is generated by training on a set of images which belong to the same class as the image to be coded. The rate distortion curves (Mean Squared Error Vs. Compression Ratio) are shown in Fig. 3 for both VQ and MVQ. As can be seen from the figure, the MVQ performance compares very closely with that of VQ and for larger compression ratios, it even performs better. The computational burden of computing the DCT of the codebook entries at encoding and decoding is a constant term. For large images, the constant term for decoding becomes negligibly small. For large images, the decompression of MVQ coded images is much better than JPEG/DCT compressed images. The future research in MVQ is to address multispectral image compression and also to study the decoding time performance for large images and compare the results with JPEG/DCT compression technique.

5. ACKNOWLEDGMENT

The authors wish to thank Kavitha Havanagi for programming support.

6. REFERENCES

1. M. Manohar and J. C. Tilton. "Progressive vector quantization of multispectral image data using a massively parallel SIMD machine," *Proc. of the Data Compression Conference*, pp. 181-190, Snowbird, Utah, March 24-27, 1992.

2. W. Pennebaker and J. Mitchell. *JPEG Still Image Data Compression Standard*, pp. 29-64, Van Nostrand Reinfold, 1993.

3. Y. Linde, A. Buzo and R. M. Gray. "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, Vol. COM-28, pp. 84-95, 1980.

4. T. Kohonen. "The self-organizing map," *Proc. of the IEEE*, Vol. 78, No. 9, pp. 1464-1480, 1990.

ORIGINAL PAGE IS
OF POOR QUALITY

5. P. A. Chou, T. Lookabaugh and R. M. Gray. "Optimal Pruning with Application to Tree-Structured Source Coding and Modeling," *IEEE Trans. on Information Theory*, Vol. IT-35, pp. 299-315, 1989.

6. A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*, pp. 309-406, Kluwer Academic Publishers, 1992.

7. M. Manohar and J. C. Tilton. "Progressive vector quantization on a massively parallel SIMD machine with application to multispectral image data," submitted to *IEEE Trans. on Image Processing*.

8. M. Manohar, J. C. Tilton, B. Kobler and P. C. Hariharan. "Model based vector quantization," *Proc. of the Data Compression Conference*, p. 497, Snowbird, UT, March 29-31, 1994.

9. B. Chitprasert and K. R. Rao. "Human visual weighted progressive image transmission," *IEEE Trans. on Communications*, Vol. 38, No. 7, pp. 1040-1044, July 1990.

10. J. Ziv and A. Lempel. "A universal algorithm for sequential data compression," *IEEE Trans. on Information Theory*, Vol. IT-23, pp. 337-343, 1977.

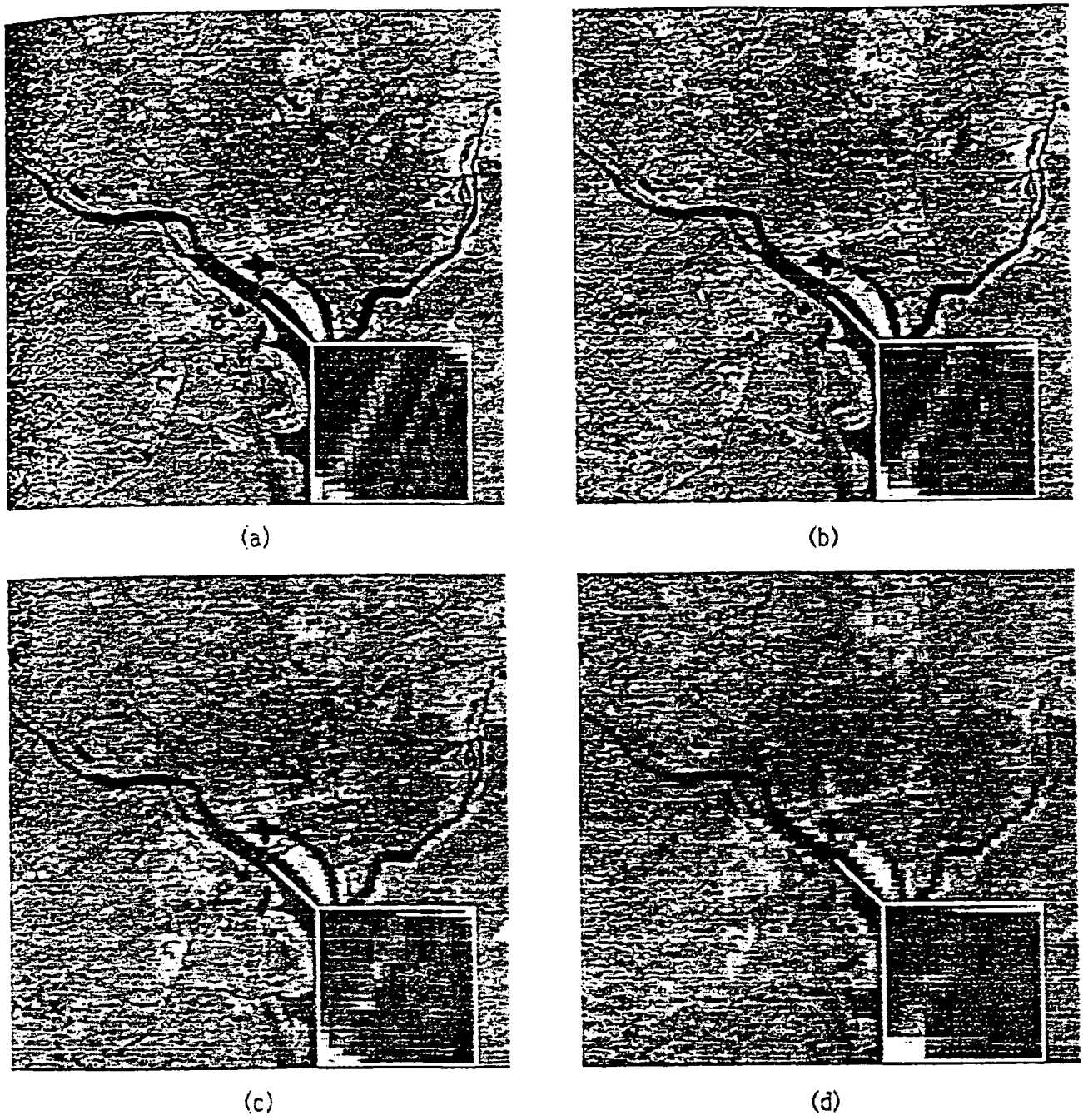

Figure 2. MVQ results on a 512x512 pixel Landsat Thematic Mapper image from over Washington, DC. (a) Original image. (b) Result from MVQ compression and reconstruction with a 4x4 vector size (CR = 6.5, MSE = 10.5). (c) Result from MVQ compression and reconstruction with a 6x6 vector size (CR = 11.1, MSE = 26.0). (d) Result from MVQ compression and reconstruction with a 8x8 vector size (CR = 27.4, MSE = 38.7). In the lower right portion of each image a 20x20 pixel section of the image from the area of the 14th Street Bridge over the Potomac River is blown up by a factor of 8.

ORIGINAL PAGE IS
OF POOR QUALITY

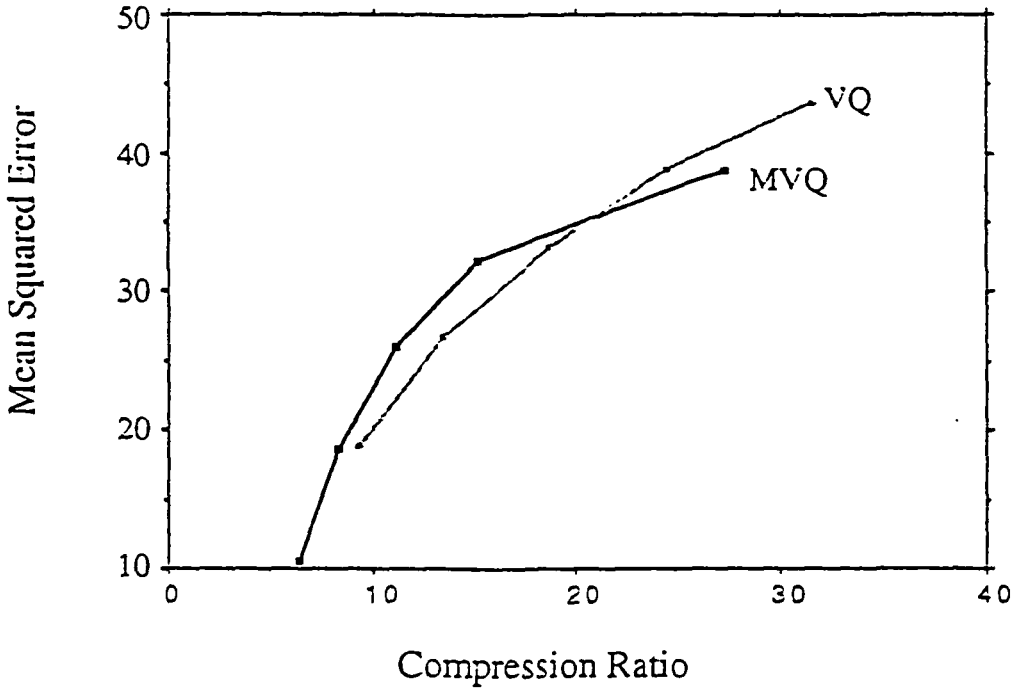

Figure 3. Rate-distortion curves for VQ and MVQ