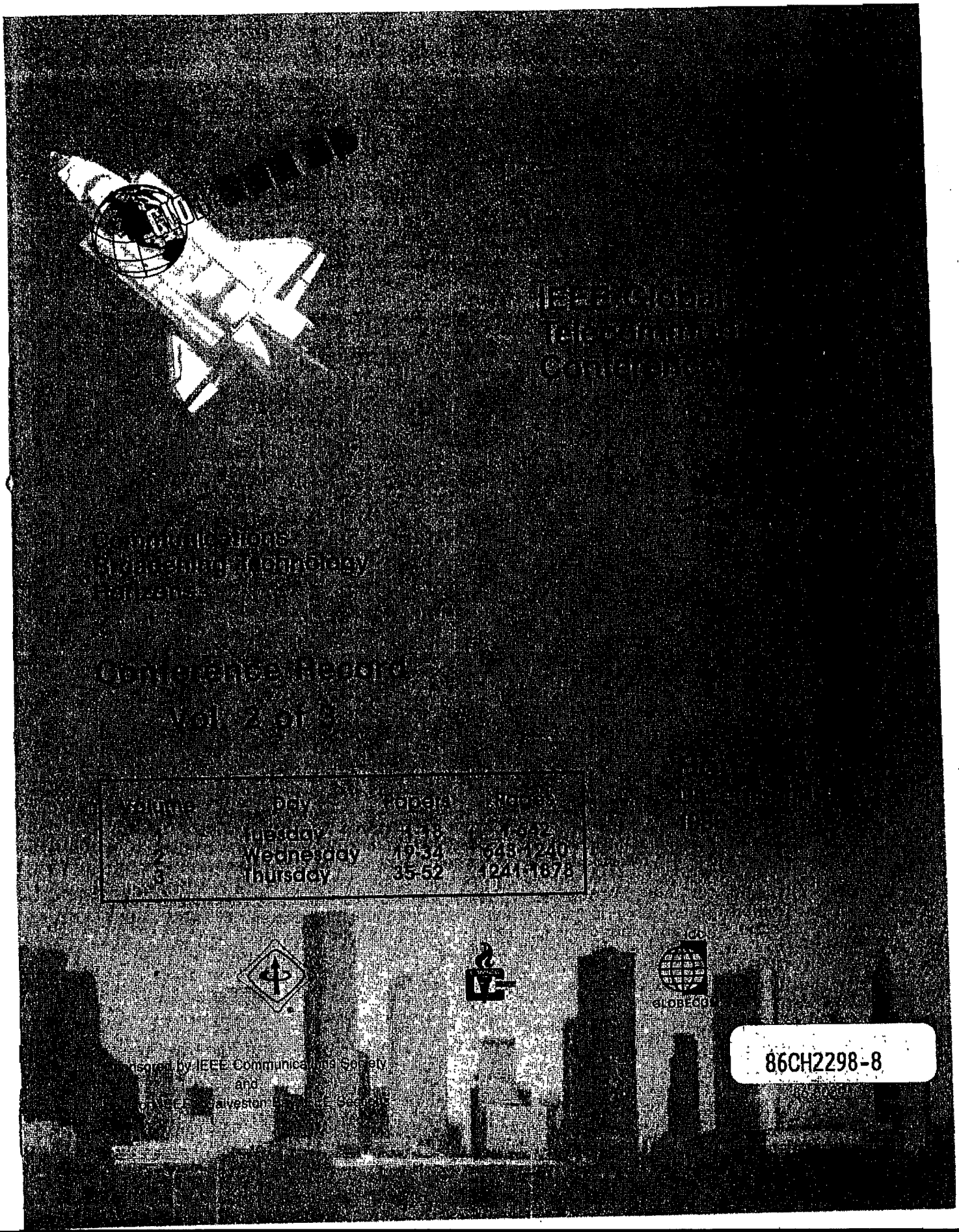
EXHIBIT L

TO DECLARATION OF S. MERRILL WEISS IN SUPPORT OF PLAINTIFF ACACIA MEDIA TECHNOLOGIES CORPORATION'S MEMORANDUM OF POINTS AND AUTHORITIES IN OPPOSITION TO ROUND 3 DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY UNDER 35 U.S.C. § 112 OF THE '992, '863, AND '702 PATENTS; AND SATELLITE DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY OF THE '992, '863, AND '720 PATENTS



Vector DPCM: Vector Predictive Coding of Color Images

Charles W. Rutledge

AT&T Bell Laboratories
Holmdel, New Jersey 07733

Abstract

Vector DPCM coding is a method of image coding that combines vector quantization coding with DPCM coding. Through use of a DPCM type prediction method and vector quantization of the differential signal, many of the problems inherent in vector quantization are alleviated. Yet, bit rates typical of vector quantization processes are attained. The results demonstrate that the typical vector quantization problems such as blocky appearing images and picture dependent codebooks are significantly reduced.

Introduction

The Vector DPCM coding method is a cross between DPCM coding and Vector Quantization (VQ) coding. DPCM is a process that takes advantage of the high correlation between adjacent pixels of a picture. Thus it uses previously processed pixels to predict a new pixel and then sends the differential value of that pixel as compared to the original. Because the differential values for each pixel are generally much smaller than pixel values in the original picture, these values may be quantized, coded, then transmitted or stored with fewer bits per pixel than the original. Vector quantization coding is a block coding technique by which blocks of signal samples, typically audio or image signals, are approximated as a group by an entity known as a vector selected from a finite set or library of vectors. The result is that each original picture block is replaced with a vector code that represents a block found in the vector codebook library that is the most similar to the original block.

The Vector DPCM process is a method by which each block of a picture is predicted from bordering pixels found in previously processed blocks. After each block is predicted it is then compared to the corresponding block in the original picture. A differential block is thus created from the original and predicted blocks and this block is then vector quantized. The vector quantization codes are then transmitted, decoded, and combined with a prediction at the receiver in a similar manner as is typical with a DPCM process. Because there are many more areas of a picture where the prediction is good and the differential signal is low, these low error codes are sent much more often than the high error codes, so the bit rate can be cut even more by implementing some type of Huffman coding. This method is relatively simple and very effective. The results have been excellent to fair with bit rates of 1 to 0.3 bits per pixel and entropies of 0.8 to 0.2 bits per pixel. This coding procedure is also so straight forward that that one could conceive of a simple inexpensive hardware implementation.

Vector Quantization has a long history, but has just recently been picking up momentum as a viable method of coding images. The best general reference for basic vector quantization is written by Gray[1] that outlines the basics for vector quantization and describes various applications and details of more advanced vector quantization systems. A good theoretical reference on vector quantization was done by Gersho [2]. He discusses various mathematical properties of vector quantization. The more recent developments using vector quantization are predictive schemes where by some type of prediction is made to remove signal redundancy followed by vector quantization. Baker and Gray introduced a method where the mean of each image block was removed and the residual signals were then vector quantized [3][4]. Cuperman and Gersho introduced adaptive differential vector coding of speech [5] where a speech waveform was partitioned into blocks of vectors and coded. A vector linear prediction is performed on the blocks to provide a difference signal to be encoded by the vector quantizer. In this scheme each frame of speech is classified into one of a number of statistical types thus providing for the selection of one of a number of fixed predictors and one of a number of quantizers. A more sophisticated predictive VQ image coding scheme has been presented by Hang and Woods [6] which involves a simple prediction and a tree structured vector quantization. Although their coding scheme

CH2298-9/86/0000-1158 $1.00 © 1986 IEEE

has an impressive coding performance, it requires numerous computations.

Vector DPCM is a new method of predictive vector quantization that provides a more efficient structure than previous methods and provides image quality similar to or better than the methods mentioned above. The background, method, and experimental results, of Vector DPCM are to follow in this paper. Section 2 discusses background issues concerning DPCM and Vector Quantization. The method and some attractive features of Vector DPCM are described in Section 3. Section 4 documents Vector DPCM results on monochrome images and Section 5 presents the results on color images.

DPCM and Vector Quantization Coding

DPCM is a predictive method of coding where a pixel is predicted using past pixel information. The prediction is then compared to the original, and the differential error or a quantized differential error is then transmitted. If the quantizer is not included in this process the picture will be reproduced exactly at the decoder. In a typical DPCM technique the pixels are processed in a raster format, and a prediction will be made as the average of the decoded values of pixels above and behind the pixel to be encoded. This value is then subtracted from the original pixel value providing the differential value of that pixel. The differential value of a pixel is most often a much smaller value than the value of the pixel itself due to the fact that adjacent pixels are highly correlated for most images. The differential value is then transmitted. It is also added to the value of the prediction to produce the coded value of that pixel that is then input to the predictor for use in predicting pixels that have yet to be processed.

The decoder is fairly simple and is actually duplicated from part of the encoder. The quantized differential pixel is received and added to the prediction, resulting in the value of the pixel that is to be displayed. This value is also input back into the predictor again for use in predicting future pixels. The quantizer is used to limit the number of possible levels of differential values that are created. The quantized values are then input into a coder that will substitute a code number for the differential value. The code number is then replaced by the differential pixel value at the decoder. Because small differential values will be transmitted much more often than large values, a Huffman coding or similar coding method can be used to code the quantized differential pixel values before transmitting them, thus further reducing the bit rate.

Vector Quantization is a method for encoding signals, typically audio or image signals, by which blocks of signal samples are approximated as a group by an entity known as a vector, selected from a finite set or library of vectors. The process depends upon creating a codebook of vectors through identification of a set of possible blocks that are in some way representative of a picture or pictures. Creating such a library of vectors is a problem that has been addressed by others, e.g., Gray [7] and Equitz [8]. In the vector quantization encoding process, the picture or data to be coded is broken up into blocks or data vectors. Each block to be coded is compared to codebook vectors and is matched up to a vector code that most nearly approximates the data contained in that particular data vector. Finally, the number that identifies the vector code found to be most similar is transmitted in lieu of the original block of data. The vector quantization decoder is relatively simple and is accomplished through table lookup. The receiver will have an identical copy of the vector codebook, and as it receives the code numbers in a predictable, sequential order, it replaces the vector codes with blocks of information as defined by the codebook and the picture is reconstructed.

Vector DPCM

The Vector DPCM method is actually a combination of DPCM and Vector Quantization coding. This method processes a picture block by block as opposed to pixel by pixel as is typically done in DPCM coding. Each block is predicted and compared to the original picture block. The difference between the predicted and original blocks is then vector quantized and vector codes transmitted. At the decoder the blocks are again predicted and the difference vectors as represented by the transmitted vector codes are added to the prediction to form a reconstructed block.

The prediction of each block is made using the edge border pixels of the block above it

and to the left of it. These blocks are used because they are previously processed blocks, since the picture is processed from left to right and from top to bottom. Each pixel of a block is predicted by using the average of the pixel above and to the left of the pixel to be predicted. Thus each pixel of a block is predicted in a raster scan type format. If the pixel above or to the left of the pixel to be predicted is a member of a previously processed block the actual coded value is used in the averaging, otherwise the pixel will be a member of the block being presently processed and only the predicted value is used. The result is that the entire block is predicted using some combination of the values of the edge pixels of the above and behind previously processed blocks (Figure 1).

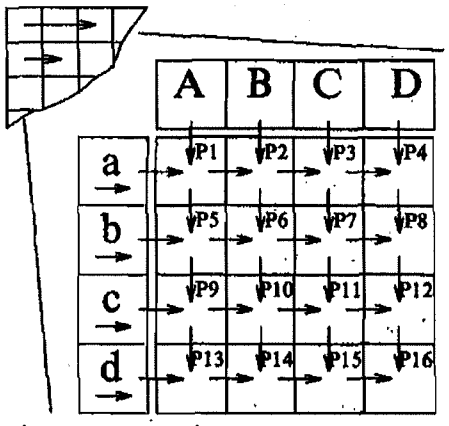


Figure 1. Method of Vector DPCM Prediction

One of the advantages of this format is its simplicity and its easy translation into a hardware or software implementation. The picture and blocks are both processed in a standard raster scan format, the prediction itself only involves an addition and a shift.

This prediction is then subtracted from the corresponding block taken from the original picture, and a difference block is created that represents the error between the two blocks. This difference block is then vector quantized, and a vector code is transmitted that is representative of a vector whose values are the most similar to the values of the difference block. Simultaneously, the value of each segment of the vector is added to the value of the prediction of each corresponding pixel to form the coded block, the same block as will be reconstructed at the decoder in the absence of transmission errors. This coded block is then input into the predictor for use in the prediction of the blocks in the picture that have not yet been processed. Since the predictor only uses the bottom and right pixels of the coded block, these are the only pixels that need to be processed and input into the predictor. This can speed up both software and hardware operations.

The decoder is very simple, can be constructed inexpensively, and will operate very fast. The decoder also operates as part of the encoder as with typical DPCM coding. If designed in VLSI, both could be designed on the same chip simply sharing similar parts. At the decoder the vector code is received and each element of that vector is added to the prediction of the corresponding pixel created in the same manner as in the encoder. The result is a reconstructed block that is displayed and also input back into the predictor as is done in the encoder. This combination of DPCM and Vector Quantization provides bit rates equivalent to that of Vector Quantization, but with much higher quality. Thus this method provides a bit rate advantage over DPCM and a quality advantage over typical Vector Quantization.

Because the blocks formed from the difference between the predicted and actual block contain much more general features than the picture itself, a codebook can be created that is much more robust and will be useful for a large variety of pictures. A normal Vector Quantization codebook must represent not only flat areas and different types of edges, but also the various degrees of light to dark areas. A codebook for differential blocks only needs to contain patterns to represent flat areas and various edges, the types of areas that are found in most all pictures. Also, because the prediction is formed using all the processed pixels immediately above and to the left of the block being coded, the prediction will tend to locate many edges, thus maintaining small differential errors and avoiding excess stress upon the codebook. Part of the advantage of this method is that it avoids the strict codebook dependency of typical Vector Quantization. Also, because a predicted

picture is usually combined with a small difference error, there is much less of the blockiness seen in the reconstructed picture than is often observed in typical vector quantized pictures. Slight blockiness was observed in some low spatial resolution pictures (256x256). The results will be discussed in greater detail in the following sections.

Another positive aspect of the Vector DPCM method is that it is simple, and thus should result in small, inexpensive hardware. The process is relatively simple as compared to many other coding techniques, especially those that operate in these bit rates. A Vector Quantization code selection process could be placed on a chip, providing the least square error calculation, error comparison function, and code generation. The other main function of DPCM could be placed on another chip. On top of these two chips all that would be required is some RAM, a couple of ROM's, and the hardware required for control, thus making a fast, compact, and inexpensive encoder and decoder.

As mentioned before, because codes representing particularly small differential values are transmitted much more often than others, a Huffman type coding technique might be incorporated into the system. Such an effort could range from using Huffman coding in a traditional form to simply transmitting a special short word if the prediction is close enough not to merit sending a code. This approach obviously would add complexity but would lower the bit rate if that is an all important goal.

Experimental Results on Monochrome Images

The initial work involved making a comparison between Vector DPCM and Vector Quantization, on similar terms. This first comparison was done on a monochrome 256x256 picture of a head and shoulder portrait of a young woman, consisting of 256 grey levels. The comparison involved two processes, vector quantization and Vector DPCM, and three slightly different methods of creating the vector codebook. A codebook was designed for each process using the LBG algorithm suggested by Linde, Buzo, and Gray [7]. For vector quantization, the LBG algorithm was used on blocks from the original picture. For Vector DPCM, the LBG algorithm was first used to create a codebook from the differential error picture that was developed by using non-quantized differential error feedback to the predictor (Method A). The final method used the LBG algorithm, but incorporating a codebook created from a differential picture that is reconstructed upon each iteration of the LBG algorithm (Method B). This picture is processed each iteration using the codebook created during the previous iteration, thus using coded picture blocks as feedback into the predictor. Each process operated using 256 codes each consisting of 4x4 blocks. The original comparison involved: 1) VQ with LBG codebook, 2) Vector DPCM with Method A codebook, and 3) Vector DPCM with Method B codebook. Both of the Vector DPCM methods provided very good results, and both were much better than the vector quantization as would be expected. The second Vector DPCM method provided the best results because the codebook was created using differential values generated from predictions made with quantized image data (Method B). The codebook created using method A used differential values generated from predictions made with the original image data.

With results of coding with a dependent codebook proving attractive, a codebook was developed that would be more robust, less dependent on the picture. This codebook was developed by using 4 different pictures as input. All four of these pictures were 256x256 monochrome images. Two were faces because it was felt that facial features were very important, one was an outdoor scene that contained many edges, and finally one was a picture of marbles that contained edges and different shadings. Using this codebook on the same portrait as used previously also provided good results. As might be expected this picture was not as good as the pictures created using the dependent codebooks, but it was still much better that typical vector quantization. The results of the processes described are found in table 1. Figure 2 is a enlarged (96 pixels x 96 pixels) portion of the vector quantized image. Figure 3 is a enlarged portion of the Vector DPCM coded image processed using a dependent codebook created using Method B. The general independent codebook was also used in processing two 512 x 512 images, Lena and a F16. These two

Table 1
Womans Portrait 4x4 Blocks

Method	Codebook	Codes	Bits/Pel	Sq Error/Pixel	Visual Quality
Vector Quantization	Dependent	256	0.5	23	Fair
Vector DPCM	Dependent	256	0.5	17	Very Good
Vector DPCM	Dependent w/feedback	256	0.5	13	Very Good
Vector DPCM	Independent	256	0.5	23	Good
Vector DPCM	Independent	512	0.56	19	Good

significantly different pictures were both processed using the 256 code independent codebook described above. The results were excellent showing great improvement over a processed 256 x 256 image because of the larger spatial resolution of image features.

The results are obviously dependent upon the codebook and the method by which it is created. The most successful method appears to be Method B described above. A comparison was made between 3 different methods of creating a codebook: 1) Equitz NN algorithm, 2)LBG (Method A), and 3) LBG with feedback (Method B). An independent codebook consisting of 256 codes was created with each method and used in the Vector DPCM process on the portrait of the woman mentioned earlier. A sample of results are found in the following table:

Method	Sq Error/pixel
Equitz NN	31
LBG	25
LBG w/feedback	23

The Equitz NN algorithm [8] requires the least computation time and the LBG with feedback or Method B requires the most but because we are creating one general codebook the time factor is not a big problem.

Experimental Results on Color Images

The color images used in this investigation were all 512 x 512 24-bit RGB images. The images were converted to the YUV components [9] before they were processed. The Y component represents the luminance of a color pixel, the U and V components are the two chrominance components and have a spatial bandwidth of about 1/5 of the Y component. Each of these three components were coded separately, then brought back together and converted back into RGB. Because the U and V components have a lower spatial bandwidth [9] they can be spatially subsampled without deteriorating overall coding performance, thus achieving lower bit rates than coding each chrominance component at the highest spatial resolution. When the U and V components are decoded, a bi-linear interpolation is done between the decoded pixels to fill in the rest of the image. The sub-sampling is done by simply taking each n-th number pixel in both the x and y directions and placing them together into vector blocks of the same size as Y, to form the image to be coded. Of course this image will contain $1/n^2$ as many vector blocks. Both the U and V components are sub-sampled, and the image is then coded in the usual Vector DPCM manner, but of course only sending $1/n^2$ as many codes for each of the U and the V components as are sent for the Y component. After the codes are transmitted and decoded the U and V components must be enlarged to their original size. A bi-linear interpolation is performed for each of the four pixels that form a sub-sampled block, the lower left hand pixel being the lower left hand pixel of the block itself.

This method of coding was performed on the picture using a variety of block sizes, codebook sizes, and sub-sampling rates. The pictures were coded using one codebook for each of the three components. The Y codebook was created using Method B, and the U and V codebooks were created using the Equitz NN algorithm in order to save time. Table 2 shows the results of a picture of a woman (Lena) coded using various size blocks. Table 3 is a comparison of a number of different pictures all coded with the same codebooks demonstrating the independence of the codebook. Figures 4 and 5 are monochrome versions of Lena and Peppers as described in Table 3. The results were all very good using the same set of codebooks created in the manner described earlier. The image of

Table 2
Lena 256 Vector Codes Various Block Sizes Independent Codebooks.

Block Size	Sub-Sampled	Sq Error/Pel	Bits/Pel	Entropy	Visual Quality
3x3	3-1	52	1.08	.83	E
4x4	2-1	72	.75	.574	VG
4x4	4-1	87	.56	.447	VG
6x6	3-1	133	.29	.20	F

Table 3
Various pictures 256 Vector Codes 4x4 Block Size 2-1 Sub-Sampled Chrominance.

Picture	Sq Error/Pel	Bits/Pel	Entropy	Visual Quality
Lena	63	.75	.574	VG
F16	95	.75	.575	VG
Sailboat	182	.75	.64	VG
Peppers	127	.75	.583	G

Table 4
Picture of a woman 256 Vector Codes Various Block Size

Block Size	Codebooks	Sq Error/Pel	Bits/Pel	Entropy	Visual Quality
3x3	3 Method B	54	1.08	.83	VG
3x3	2 Method B	57	1.08	.81	VG
3x3	3 Method A	54	1.08	.83	VG
4x4	3 Method B	74	.75	.599	G
4x4	3 Method A	78	.75	.544	G

peppers had some noticeable distortions at the edges between the bright green peppers and the bright red peppers. The other images showed very little distortion. Table 4 presents some results from a 256x256 color image including a image in which only two codebooks were applied, one for Y and a common one for U and V.

Discussion

The image quality can probably be slightly improved in a couple of ways. The LBG algorithm Method B has consistently provided as good or better results than the other methods tried. This method of codebook generation was used only on the Y component codebook while the Equitz NN algorithm was used for generating U and V codebooks. Thus it can be expected that by using the LBG Method B algorithm to generate these U and V codebooks there will be at least a slight improvement. Also the codebooks were all created using four pictures as described earlier. The codebooks may be improved by using a different set of pictures or a larger variety of pictures. Thus, the proper selection of pictures used to generate the codebook is not clear.

An on-going project involves implementing this algorithm in a variable block size coding scheme. This would involve using 8x8 blocks in non-detailed areas, 4x4 blocks in more detailed areas, and 2x2 blocks in the highly detailed areas. This algorithm would add slightly more complexity but would also be oriented toward varying the amount of information depending upon the complexity of the image. It would provide a more consistent quality across the entire image and also avoid sending too much unnecessary information in non-detailed sections of the image.

This algorithm also may prove useful in providing images for Videotex applications. Thus research should be done on processing 256x200 images, and on using only 192 vector codes so that they might easily fit into the NAPLPS structure. Also other features of NAPLPS that might provide the starting values for the top and left edges should be examined.

Acknowledgements

I would like to thank Hsueh-Ming Hang and Joe Othmer for numerous valuable discussions and Bill Ninke for his support of this work.

References

[1] Gray R. M. "Vector Quantization", *IEEE ASSP Magazine*, pp. 4-29, April 1984.

[2] Gersho A. "On the Structure of Vector Quantizers", *IEEE Trans. on Information Theory*, Vol. IT-28, No. 2, pp. 157-166, March 1982.

[3] Baker R. L. and Gray R. M. "Image Compression Using Non-Adaptive Spatial Vector Quantization", in *Proc. 16th Asilmar Conf. Circuits, Systems & Computers*, pp. 55-61, October 1982.

[4] Baker R. L. and Gray R. M. "Differential Vector Quantization of Achromatic Imagery", *Proc. International Picture Symposium*, pp 105-106, March 1983.

[5] Cuperman V. and Gersho A. "Adaptive Differential Vector Coding of Speech", *IEEE Global Telcomm Conf*, pp. 1092-1095, November 1982.

[6] Hang H.-M. and Woods J. W. "Predictive Vector Quantization of Images", *IEEE Trans. on Communications*, Vol. COM-33, No. 11, pp. 1208-1219, November 1985.

[7] Linde Y., Buzo A., and Gray R. M. "An Algorithm for Vector Quantizer Design", *IEEE Trans. on Communications*, Vol. COM-28, No. 1, pp. 84-95, January 1980.

[8] Equitz, W. H. "Fast Algorithms for Vector Quantization Picture Coding." M.Sc. thesis, Dept. Elec. Eng. and Comput. Sc., MIT, MA, June, 1984.

[9] Limb J. O., Rubinstein C. B., and Thompson J. E. "Digital Coding of Color Video Signals -- a Review", *IEEE Trans. on Communications*, Vol. COM-25, pp. 1349-1385, November 1977.

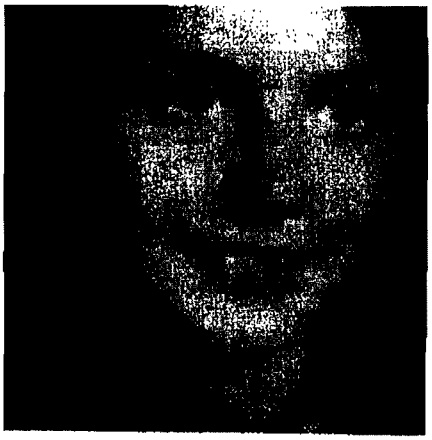


Figure 2 Vector Quantization

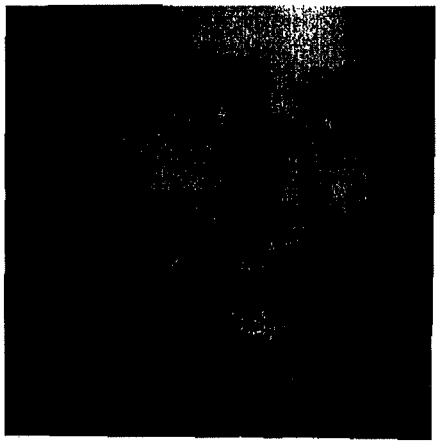


Figure 3 Vector DPCM

Figure 4 Lena

Figure 5 Peppers