Vector quantization

Julio A. Hernández
Associate Professor, PUPR
Department of Electrical Engineering

Abstract

Data compression is the packing of data, the process of transforming a body of data into a smaller representation from which the original or an approximation of the original can be computed at a later time. Most data sources contain redundancies such as non-uniform symbol distribution, pattern repetition, and positional redundancy. A data compression algorithm encodes the data to reduce these redundancies.

Vector quantization (VQ) is an efficient data compression technique for speech and images. VQ maps a pattern defined by a discrete vector into a short digital sequence suitable for transmission over a digital channel or storage in a digital medium. Today, with the computing power available, the time spent for data processing has been reduced and the application of vector quantization for real-time coding is once again considered. A study of different techniques of vector quantization will be done. The details of some schemes will be presented.

Quantización vectorial

Sinopsis

La compresión de datos es la acción de codificar un grupo de datos, el proceso de transformar un grupo de datos en forma compacta a su forma original o una aproximación que pueda recuperarse nuevamente. La mayoría de las fuentes de información contienen redundancias en forma no uniforme, patrones repetidos y redundancia posicional. Los algoritmos de compresión de datos reducen estas redundancias.

La quantización vectorial (QV, por sus siglas en inglés) es una técnica de compresión de datos eficiente para voz e imágenes. La QV toma muestra de los patrones definidos por vectores discretos y los convierte en secuencias digitales cortas adecuadas para transmitirlas por medio de un canal digital o para almacenarlas en forma digital. Con la capacidad de computadoras que hay hoy el tiempo para procesar datos ha disminuido considerablemente y la técnica de QV se considera para para codificar a tiempo real. En este trabajo se estudian diferentes técnicas de QV y se presentan los detalles de algunos de los esquemas serán presentados.

Basic quantization and coding

Memoryless quantizer

The zero memory quantizer which operates on one input sample at a time, and the output value depends only on that input. This type of quantizer is useful in image coding techniques. Unfortunately the quantization operation is irreversible because the quantization maps a range of values with one quantization level. Hence the input value cannot be reproduced exactly as the original. Depending on the application the distortion introduced by the quantization can be adjusted. There are several quantizer designs available that offer various trade-offs between picture quality and complexity.

The Lloyd-Max quantizer

This scalar quantizer minimizes the mean square error for a given number of decision levels (Jain, 1989). Let **u** be a real scalar random variable with a continuous probability density function $P_u(\mathbf{u})$. If it is needed to find the quantization levels t_k and the inverse quantization levels ξ for an L-level quantizer

$$\epsilon = E[(u-u^*)^2] = \int_{tl}^{t_L+1} (u-u^*)^2 P_u(u)du \qquad (1)$$

is minimized. Rewriting this becomes

$$\epsilon = E[(u-u^*)^2] = \sum_{i=1}^{L} \int_{tl}^{t_L+1} (u-r^*)^2 P_u(u)du \qquad (2)$$

From basic calculus, it is known that the minimum occurs where the derivative with respect to t_k and r_k is zero. This gives

$$\frac{\partial \epsilon}{\partial t_k} = (t_k - r_{k-1})^2 P_u(t_k) - (t_k - r_k)^2 P_u(t_k) = 0 \qquad (3)$$

$$\frac{\partial \epsilon}{\partial t_k} = \int_{tl}^{t_k+1} (u-r_k) P_u(u)du = 0 \ , \ 1 \le k \le L \qquad (4)$$

Using the fact that $t_{k-1} \le t_k$, simplification of the preceding equations gives

$$t_k = \frac{r_k + r_{k-1}}{2} \qquad (5)$$

and

$$r_k = \frac{\int_{t_k}^{t_k+1} uP_u(u)du}{\int_{t_k}^{t_k+1} P_u(u)du} = E[u/u \in \psi_k] \qquad (6)$$

where r_k is the k^{th} interval $[t_k, t_{k+1}]$. These results state that the optimum quantization levels are located halfway between the reconstruction level, which lie at the center of mass of the probability density in between the quantization levels. Equations (5) and (6) are nonlinear equations. The equations have to be solved simultaneously given the transition interval t_1 and t_{L+1}. In practice,

these equations can be solved by an iterative numerical method.

The application of PCM for a signal is performed in the following manner:

- The signal is sampled
- All levels are quantized
- The signal is transmitted or fed to a digital modulator for transmission

Figure 1 shows the process of pulse code modulation (PCM), a digital modulator for transmission. Generally PCM is coded by a fixed-length binary code. The amount of bits required depends on the signal to be coded. For example, to code 512 different values at least nine bits are needed. NB=LogN(amount of codes)/LogN(2). NB=Log(512)/Log(2)=9. Log base 2 is commonly used.

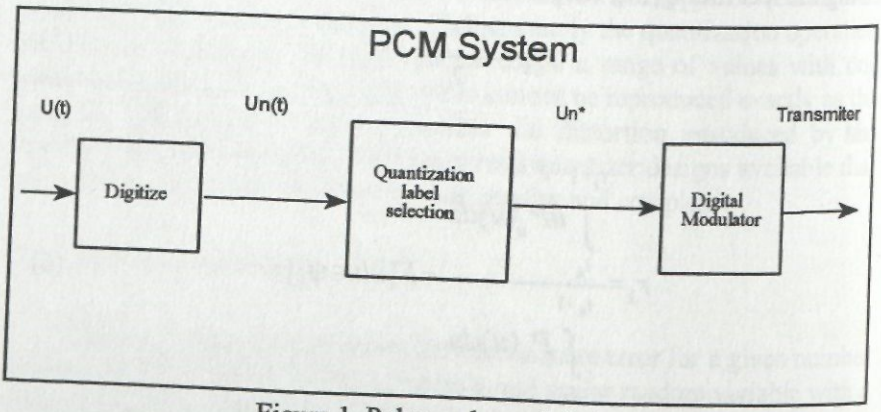

Figure 1. Pulse code modulation

Entropy Coding

If binary data are coded and the distribution of the data not uniform, then the entropy (B=average bit length) of those data points will be less than B, and

there is a code that uses less than B bits per data point. The goal of entropy coding is to encode a block of M data points containing MB bits with probabilities P_i; {I=0,1,..., L-1}, $L=2^{MB}$, by $-\log_2 P_i$ bits, so that the average bit rate is

$$\Sigma_i P_i(\log_2 P_i)=H \qquad (7)$$

This gives codes with variable-length for each block, where the blocks with higher probability of occurrence are coded by short-length codes. If $-\text{Log}_2 P_i$ is not an integer, the achieved rate exceeds H, but approaches it asymptotically as the number of blocks increase. For a finite block size, the most efficient fixed to variable length encoding method is Huffman coding [(Jain, 1989); (Barnley and Hurd, 1993)].

The Huffman coding algorithm

1 Arrange the symbol probabilities P_i as follows:

- Put the two smallest P_i together. Their sum forms a node.
- Subsequently, the next adjacent P_i has to be greater or equal to the sum of the previous node formed.

2- Make a tree:

- Merge the two nodes with smallest probabilities to form a new node whose probability is the sum of the two merged nodes.
- Arbitrarily assign 1 and 0 to each pair of branches merging into a node, but assigning 1 to the smaller of the two branches.
- Continue the process until a home is formed. All branches are

together.

- Read sequentially from the home to the symbols.

The preceding algorithm gives the Huffman Codebook for a set of symbols given their probabilities. The Huffman code is not unique. For example, let the symbols A,B,C, D and E with probabilities 0.264, 0.053, 0.108, 0.137 and 0.438 respectively.

1 - Merge {B,C} to form a tree with weight PBC=0.161. Here, the use of the notation PBC means the probability of seeing either symbol B or symbol C.
2 - Merge BC and D to form a tree with weight PBCD=0.298.
3 - Merge A and BCD to form a tree with weight PABCD=0.562.
4 - Merge ABCD and E, completing the tree.

Figure 2 shows the graphic interpretation. The code words are: A=00, B=0100, C=0101, D=011, E=1. Using Huffman coding the average length per code word is 2.02 bits. The entropy average bit length is 1.9932

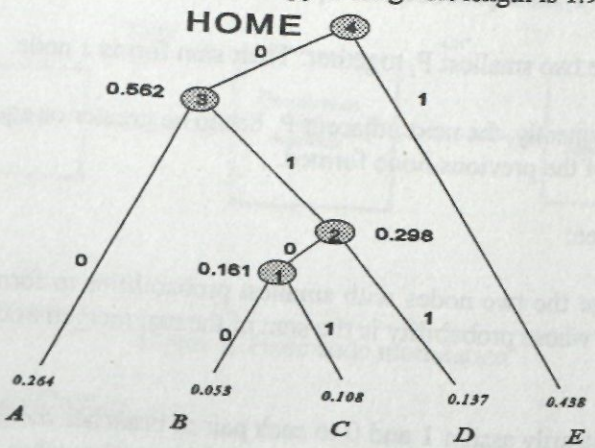

Figure 2. Huffman tree for codification of letters A B C D. Steps are indicated at the nodes

Vector quantization

Vector quantization has been established as an efficient data compression technique for speech and images. Using a suitable communication channel, VQ maps a sequence of continuous or discrete vectors into a digital sequence for transmission. The main goal is to reduce the amount of data while fidelity levels are preserved. Vector quantization has received much attention recently as a method for coding images and motion video and has been shown to be an effective method for coding images. VQ methods are a family of schemes that are being actively studied for image codification. Although simple memoryless VQ schemes yield acceptable performance at low bit rates, there are other quantization schemes that have been shown to be more efficient.

In this thesis the application of VQ to image compression will be performed decomposing the images in blocks of 4x4 pixels, using MATLAB block processing modules. First the images will be transformed into a column form, then re-order in a matrix of 16 elements column vectors. Those matrixes will be used as the input to the algorithm. For the neural networks algorithm the same matrix will be used. The neural networks input has to be a normalized vector from the re-ordered image matrix. For the conventional VQ techniques it is not necessary to normalize the input vectors, but it is recommended because the ratio of compression would increase. That is because many vectors could be quantized with the same normalized vector if their difference is only in their length, but in turn it is needed to store the length as an additional value which reduces compression.

Image quantization

After sampling an image and representing its digital values, the next step is quantization or mapping. A quantizer maps a continuous variable or vector **u** onto a discrete variable or vector **u***, which takes values from a user defined finite range $\{r_1, ..., r_n\}$, which represent quantization levels or numbers (Jain, 1989; Barnley and Hurd, 1993; Lindley, 1991). Each r contains a value that represents a group of values. This mapping is generally a staircase function, as figure 3 shows.

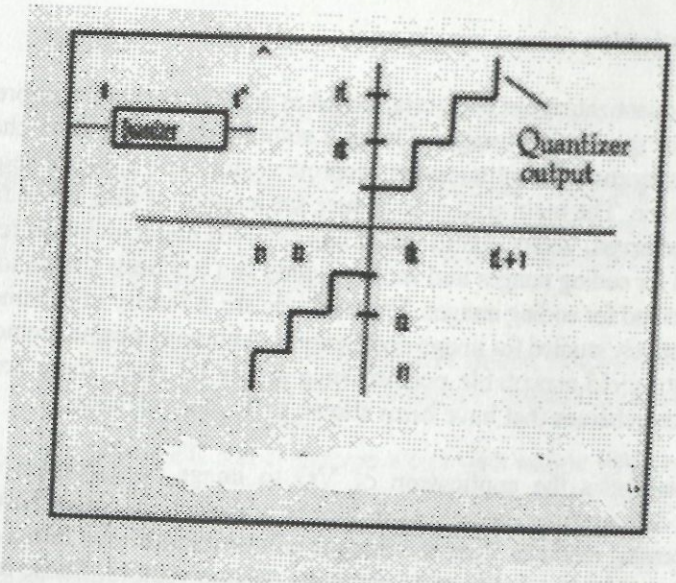

Figure 3. A quantizer (Jain, 1989)

The quantization rule is as follows: Define $\{t_k,..., t_{L+1}\}$ as a domain of different decision levels with t_1 and t_{L+1} as the lower and upper values of this domain, respectively, of the continuous space to be quantized **u**. If **u** lies in interval (t_k, t_{k+1}), then it is mapped to the corresponding map space r_k, the k^{th} reconstruction level. Figure 4 shows a basic C decoder model.

For example, the simplest and most common quantizer is the uniform quantizer, which has a uniform distribution of the levels of **u** to the levels of the map space. If the output of an image sampler take continuous values between 0.0 to 1.0 and the samples are quantized uniformly to 512 levels, then the transition and reconstruction levels are:

$$t_k=\frac{(k-1)}{512},\ k=1,2,...,513, \quad r_k=\frac{512t_k+5}{512},\ k=1,2,...,512. \qquad (8)$$

The interval $I=t_k-t_{k-1}=r_k-r_{k-1}$ is constant for different values of k.

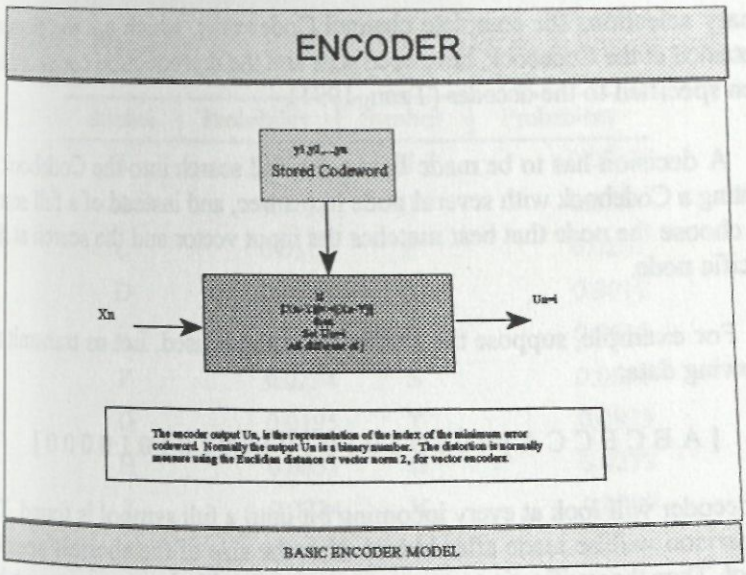

Figure 4. The basic C decoder model

Conventional vector quantization methods

Tree-search vector quantizer

Vector quantizer

A vector quantizer is a system used for coding symbols or data pattern. The basic system receives the input signals and performs a search of its Codebook ROM where stores all the pattern or symbols that represent vectors to compare with the input one, and the output of the quantizer will be the corresponding index of the Codebook of the minimun distortion codeword.

After using the Huffman coding technique to code the symbols that can be macrobloks or a sequence, the data or signal is coded using zeros (0) and ones (1), depending on the path in the tree. The codification takes place at the encoder, so at the decoder we need the inverse process for reconstruction. It has to define (R), the minimum bits that represent a symbol or world. After R

binary selections the complete channel Codeworks, which are the elements contained in the Codebook, have been sent and the reproduction codeword has been specified to the decoder (Tzou, 1991).

A decision has to be made between a full search into the Codebook or creating a Codebook with several node like a tree, and instead of a full search you choose the node that best matches the input vector and the search at that specific node.

For example, suppose the Huffman coding is used. Let us transmit the following data:

[A B C E C C A]=[0 0 0 1 0 1 0 1 0 1 1 0 1 0 1 0 1 0 0 0 0]

The decoder will look at every incoming bit until a full symbol is found. The comparison will be made after N bits. N is the size of the shortest possible symbol. Then the verification process starts again with the next incoming bit. The decoder starts at Home position, which is first bit to compare, after a symbol is successfully constructed. Table 1 shows the binary representation for A, B, C, D, and E.

Table 1. Binary representation of symbols

Symbol	Binary
A	1010
B	1011
C	1100
D	1101
E	1110

Table 2. Probability of occurrence of the 26 letters in English text

Symbol	Probability	Symbol	Probability
A	0.0761	N	0.0711
B	0.0154	O	0.0765
C	0.0311	P	0.0203
D	0.0595	Q	0.0010
E	0.1262	R	0.0615
F	0.0234	S	0.0650
G	0.0195	T	0.0933
H	0.0551	U	0.0272
I	0.0734	V	0.0099
J	0.0015	W	0.0189
K	0.0065	X	0.0019
L	0.0411	Y	0.0172
M	0.0254	Z	0.0009

Note: (Barnley and Hurd, 1993)

To transmit the given data the system must transmit 20 bits with no codification and an average length of 14 bits using Huffman coding. As a result the Huffman code algorithm performed a compression of 14/20 or reduced the data to a 70% of the original size. Figure 5 shows a graphic interpretation.

Multi-step vector quantizer

Figure 6 shows a multistep VQ, which is a tree-searched VQ where only a single small codebook is stored for each layer (node) of the tree instead of a different codebook for each node of each layer. Such codes provide the computation reduction of tree-searched codes while reducing the storage requirements below those of even ordinary VQs. The first instance of such a code was the multistage codebook (Juang and Gray, 1982). The first quantizer

is designed as in the tree-searched case. This codebook is used to encode the training sequence and then a training sequence of errors or residual vectors is formed. For waveform coding applications the error vectors are simply the difference of the input vector and their codeword. In Multistage VQ with two stages, the input vector is first encoded by one VQ and an error vector is formed. The second VQ then encodes the error vector; then the transmision will be (i,j)=Un· Un, which has the indexes of each codeword.

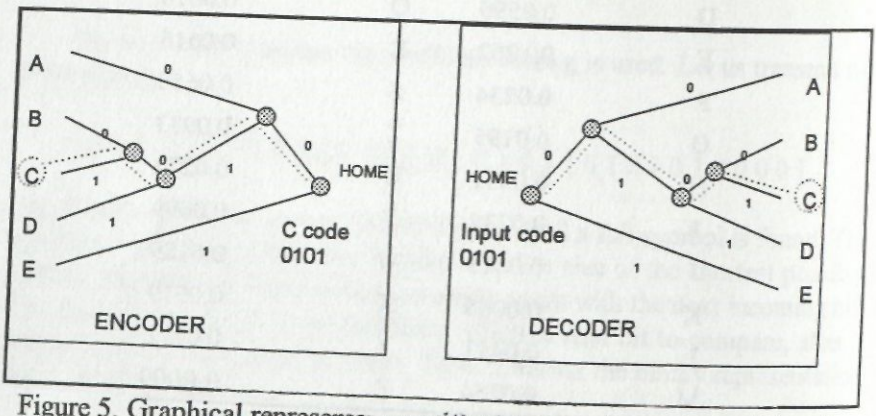

Figure 5. Graphical representation of the tree-search encoder and decoder

For example, Let X_n = [5 1 7 0 3] and suppose that in the search the best m atch was Y_2=[5 0 6 0 3]; then the first VQ output is 0010; the error is [0 1 1 0 0]. After the second search, the best match is e_1=[0 1 2 0 0], so the error VQ output is 0001. Note that at the decoder there are also a sequence codebook and an error codebook. After the decoder search, its outputs are:

$$Y_2 = [5\ 0\ 6\ 0\ 3] \quad , e_1 = [0\ 1\ 2\ 0\ 0] \tag{9}$$

the reconstruction is:

$$X_n = Y_i + e_j = Y_2 + e_1 = [5\ 1\ 8\ 0\ 3] \tag{10}$$

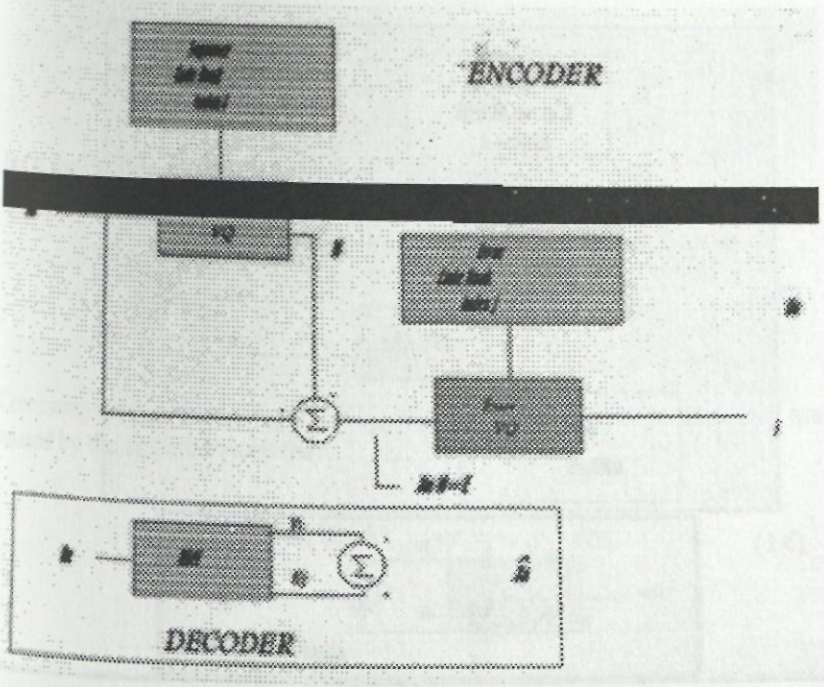

Figure 6. A multistage vector quantizer

Gain/shape vector quantizer

One example of a product-code, which is a vector quantization method that is implemented using several scalar quantizer, is a gain/shape VQ, where separate, but interdependent, codes are used to code the "shape" and "gain" of the waveform. "Shape" is defined as the original input vector normalized by removal of the "gain" term such as energy in a waveform coder or LPC residual energy in a vocoder. Gain/shape encoders were introduced by Buzo, Gray and Markel (1980) and were subsequently extended and optimized by Sabin and Gray (1982;1984). A gain shape VQ for waveform coding with a square-error distortion is illustrated in figure 7.

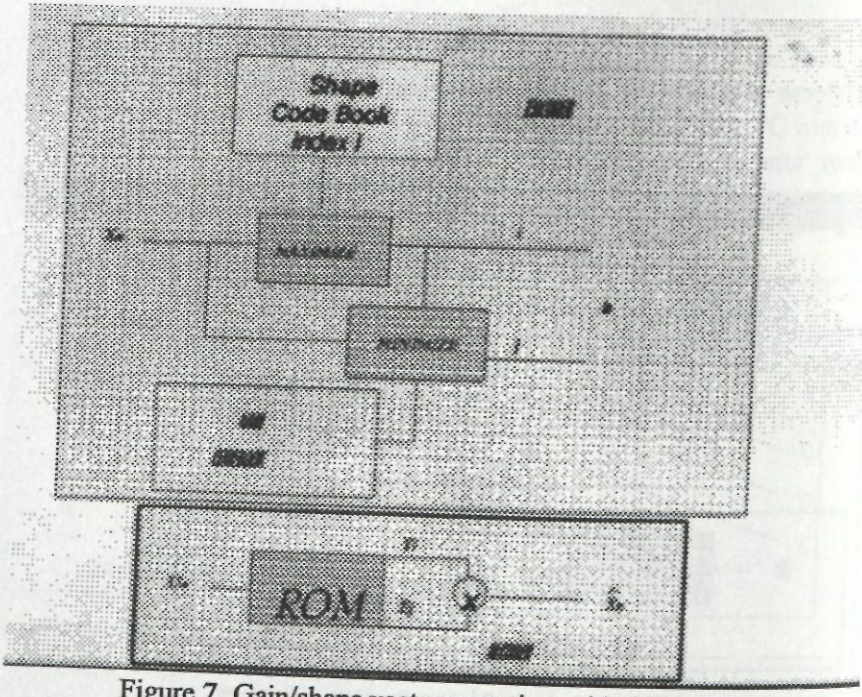

Figure 7. Gain/shape vector quantizer with the decoder.

The derivation of those formulas is as follows:

Let $X_n=[x_1\ x_2\ x_3]$ unnormalized and $Y_n=[y_1\ y_2\ y_3]$ normalized
Let XX_n be the normalized vectorX_n

$$XX_n = = \frac{\Sigma X_i}{||X_n||} \quad , \quad ||X_n|| = \sqrt{x_i + x_2 + x_3} \tag{11}$$

$$\|X_n\| = \sqrt{\sum_i x_i^2}, \quad XX_n = [x_1 x_2 x_3] \tag{12}$$

but if the Euclidean norm is considered, then the value is

$$XX_n = = \frac{\sum_i X_i}{||X_n||} \quad , \quad XX_n = \sqrt{x_i + x_2 + x_3} \tag{13}$$

Comparing the two vector, Y_m and XX_n, if they are not the same then they are related by the angle between them.

$$||X_n||.||Y_m|| \cos(\theta) = XX_{nY_m} = x_1 y_1 + x_2 y_2 + x_3 y_3 \tag{14}$$

$$\|Y_m\| \cos(\theta) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}} = X_n{}^t Y_m \tag{15}$$

to be maximized.
Let D be the square error between X_n and $\gamma_i Y_m$

$$D = (X_n - \gamma_i Y_m)(X_n - \gamma_i Y_m)^t \tag{16}$$

$$D = (X_n - \gamma_i Y_m)(X_n{}^t - \gamma_i Y_m{}^t) \tag{17}$$

$$D=X_n X_n{}^t \gamma_i Y_m - X_n \gamma_i Y_m{}^t + \gamma_i{}^2 Y_m Y_m{}^t \quad (18)$$

$$D=X_n X_n{}^t - 2\gamma_i X_n{}^t Y_m + \gamma_i{}^2 Y_m{}^2 \quad (19)$$

if $X_n=[x_1\ x_2\ x_3]$ and $Y_m=[y_1\ y_2\ y_3]$ then $X_n Y^t{}_n = X^t{}_n Y_m$,therefore,

$$x_1y_1+x_2y_2+x_3y_3 = y_1x_1+y_2x_2+y_3x_3 \quad (20)$$

See equation (19), but the distance between vectors is measured using a vector norm. Now define

$$\gamma_x=\|X\|_2 \doteq \sqrt{\sum_i X-i^2} \doteq \sqrt{x_1{}^2+x_2{}^2+x_3{}^2} \text{ also } \gamma_x{}^2 \doteq X_n{}^2. \quad (21)$$

The last term of the equation of D can be dropped because it can be easily calculated when i and m are known . The remanding equation for the distance then becomes $D=X_n X^t{}_n - 2\gamma_i X^t{}_n Y_m$ to be a minimum, or the error to be minimized. Figure 7 shows the Gain/Shape vector quantizer with the decoder.

Separating mean vector quantizer

Another example of a multistep product code is the separating mean VQ where a sample mean is removed. Define the sample mean of a k-dimensional vector. In the separated mean VQ a scalar quantizer is first used to code the sample mean of the vector, then the coded sample mean is subtracted from all the components of the input vector to form a new vector with approximately zero sample mean. This vector is then vector quantized. Such a system if

depicted in figure 8. The equations for this quantizer are as follows:

Let $X_n=[x_1\ x_2\ x_3 \ldots x_n]$ and X^*_n be the mean of X_n then

$$Y_i = \frac{(X_n - X_n^*)}{||X_n||} \tag{22}$$

Let u_j = scalar value that approaches X_n The transmited values are u that contain the values of i and j, so at the decoder the reconstruction is:

$$\hat{X}_n \approx Y_i + u_i = \frac{(X_n - X_n^*)}{\|X_n\|} \cdot \|X_n\| + X_n^* \; but \hat{X}_n \approx X_n^* \tag{23}$$

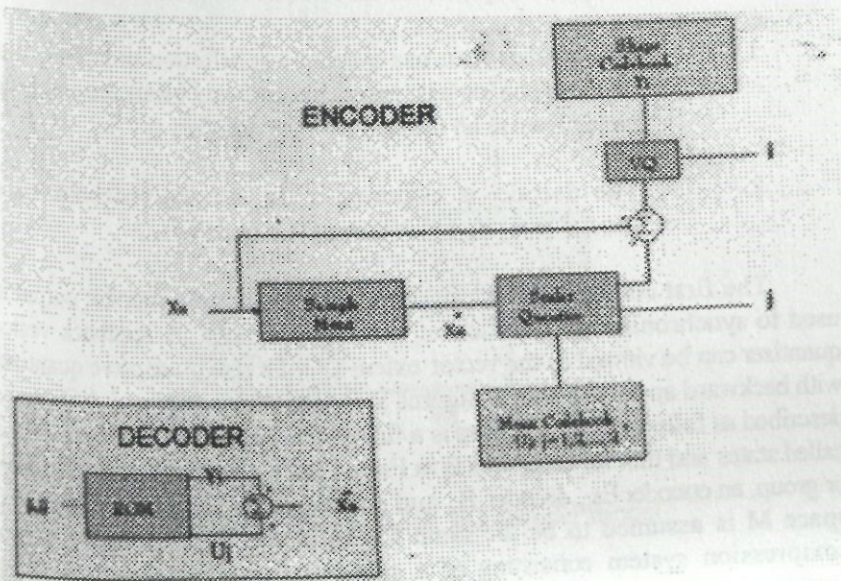

Figure 8. The separating mean VQ model

$$\hat{X}_n \approx Y_i + u_i = \frac{(X_n - X_n^*)}{\|X_n\|} \cdot \|X_n\| + X_n^* \; but \hat{X}_n \approx X_n^* \qquad (24)$$

therefore the reconstruction is $\hat{X}_n$.

The state or decision VQ has several codebooks inside. Adaptation can be incorporated into a vector quantizer by using different codebooks or separating each codebook into a set of sub-codeboks, where the codebooks are adapted on the basis of past input vectors. The decoder must have a synchronization with the encoder to know which codebook is being used in order to decode the channel symbols. This can be accomplished in two ways:

1. The encoder can use a codebook selection procedure that depends only on past encoder output and hence the codebook sequence can be tracked by the decoder.

2. The decoder is informed if the selected codebook via a special low rate side channel (Gray, 1984)

The first condition is call feedback VQ because the encoder output is used to synchronize the selection of the new codebook. A feedback vector quantizer can be viewed as the vector extension of a scalar adaptive quantizer with backward approximation as figures 9 and 10 show. Feedback VQ can be described as follows: Suppose there is a finite space Fs whose elements can be called states and that for each state fs in Fs you have a separate quantizer class or group, an encoder Es_n, decoder B_s, and codebook C_s. The channel codeword space M is assumed to be the same for all of the VQ's. Consider a data compression system consisting of a sequential machine such that if the machine is in state fs, then it uses the quantizer with encoder Es_n and decoder

B_s. Using the synchronization value it then selects its next state by a function called a state-transition function f such that given a state fs and a channel symbol x, then f(x,fs) is the new-state of the system. Specifically, given a sequence of input vectors $\{X_n;n=0,1,...,2^R\}$ and an initial state fs_0, then the sub-sequence state fs_n, channel symbol sequence x_n, and reproduction sequence x^*_n are defined recursively for n=0,1,2,...n as $x_n = E_s(X_n)$, and reconstructed as $x^*_n = B_s(x_n)$, $fs_{n+1} = f(x_n, fs_n)$. Since the next state depends only on the current state and the channel codeword, the decoder can track the state if it knows the initial state and the channel sequence. If the state space has a limited universe, then the resulting system can be called a finite-state vector quantizer (FSVQ). A disadvantage in all feedback quantizers is that channel error can accumulate and cause severe reconstruction errors. As with scalar feedback quantizer systems, this problem must be handled by cyclical resetting of the state control line.

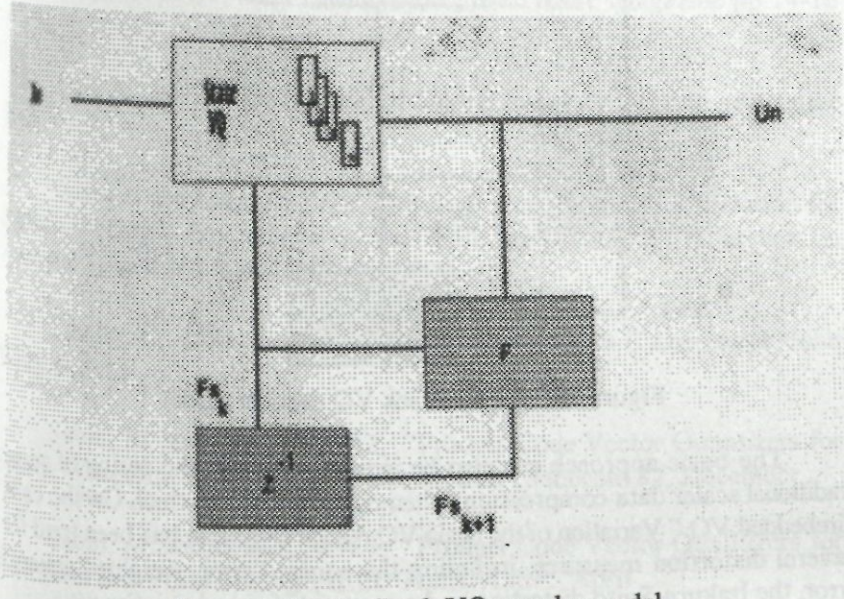

Figure 9. The feedback VQ encoder model

Conclusion

The basic Lloyd's iterative algorithm and how this method can be improved and applied to a variety of vector quantization systems, ranging from the fundamental VQ techniques to a feedback model, was described. Through a variety of examples of systems and code design simulations, some of the tradeoffs among performance, rate, complexity, and storage for these codes were illustrated.

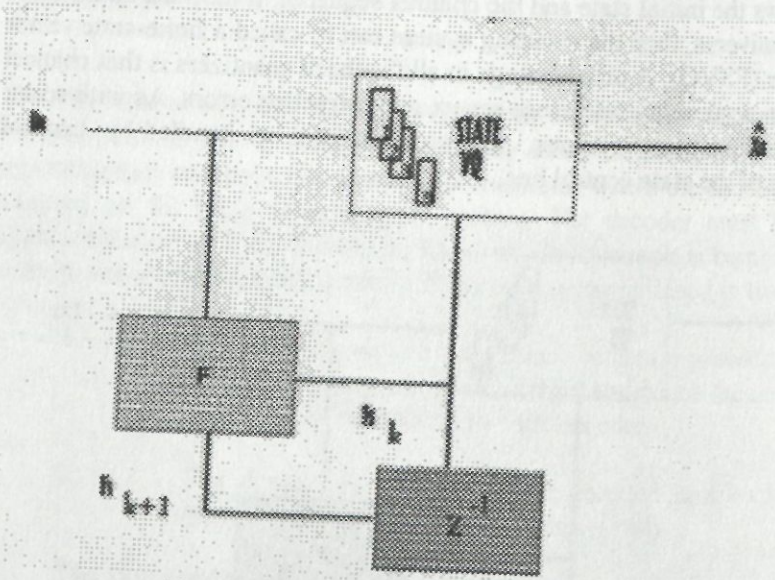

Figure 10. The feedback VQ decoder model

The basic approach can also be incorporated into the design of some traditional scalar data compression schemes, an approach which Gersho calls "imbedded VQ." Variation of the basic VQ design algorithm has been tried for several distortion measures, including the squared error, weighted squared error, the Itakura-Saito distortion and a segmented signal to noise ratio.

The theory of the VQ techniques is widely used. The complexity of the VQ technique depends on the type of application used. The result from a VQ system may vary from one application to the other. Therefore, the efficiency of your VQ system also depends on the application selected.

References

Buzo, A., Gray, A., Gray, R.., and Markel, J., 1980 "Speed coding based upon vector quantization", IEEE transactions on Acoustic Speech and Signal Processing ASSP-28 pp.562-574 (October)

Barnley, M., Hurd L.. Hurd, 1993, *Fractal Image Compression*, AK Peters

Gray, R, 1984, " Vector Quantization", IEEE ASSP Magazine pp.14-16 April

Jain, A., 1989, Fundamental of Digital Image Processing,. Precntice Hall

Juang,B.and Gray, A, 1982, "Multiple stage Vector Quantization for Speech Coding", Proceedings of the IEEE International Conference on Acoustic Speech and Signal Processing (April)

Lindley, C., 1991, *Practical Image Processing in C Language*, John Wiley & Sons

Sabin, M. and Gray, R, 1982, "Product Code Vector Quantizers for Speech Waveform Coding". Conference Reord Globecom'82, December

Sabin, M..and Gray,R., 1984, " Product Code Vector Qualltizers for Waveform and Voice Coding", IEEE Trans. ASSP., April

Tzou, K., 1991, *Vector Quantization for Images*, july 29