A RECURSIVE TECHNIQUE FOR ADAPTIVE VECTOR QUANTIZATION

Robert A. Lindsay
Unisys Corporation

ABSTRACT

Vector Quantization (VQ) is fast becoming an accepted, if not preferred method for image compression. VQ performs well when compressing all types of imagery including Video, Electro-Optical (EO), Infrared (IR), Synthetic Aperture Radar (SAR), Multi-Spectral (MS), and digital map data. The only requirement is to change the codebook to switch the compressor from one image sensor to another. However, codebooks can be difficult to design because data may not be available or may not accurately represent the pdf. This stimulates the need for an algorithm that can simultaneously design a codebook while vector quantizing the data.

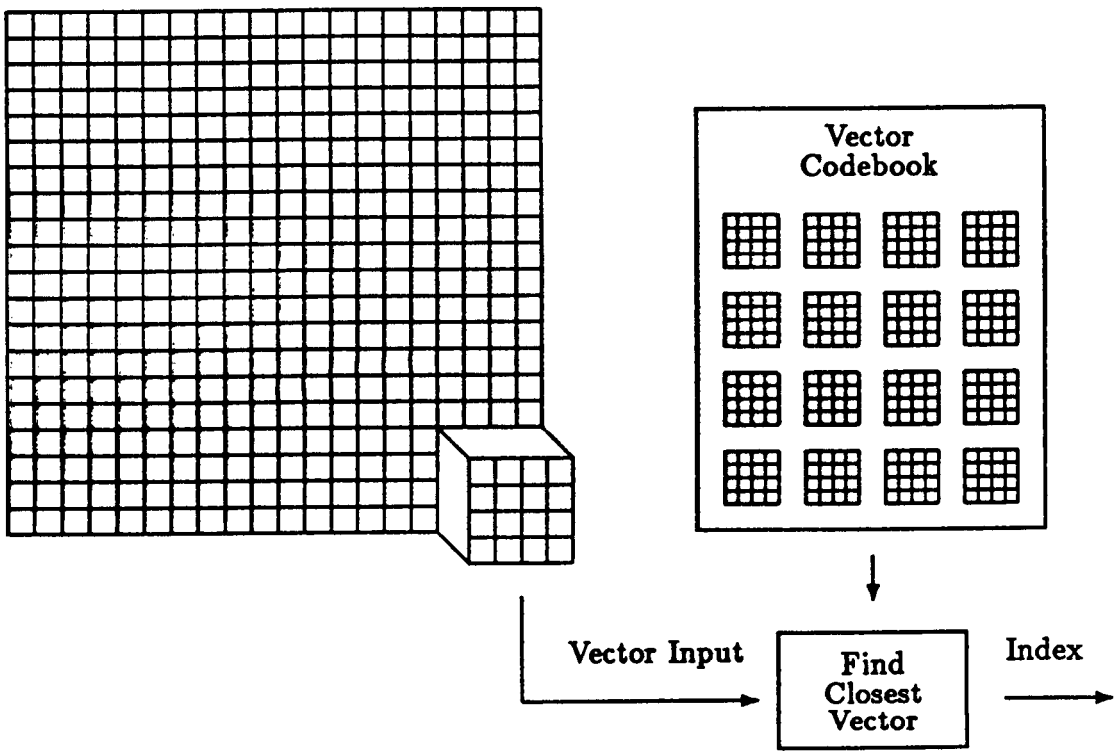
There are several approaches for designing codebooks for a vector quantizer. The most common algorithm being the LBG or generalized Lloyd. Entries in the codebook represent the centroid of the data that is associated with a respective Voroni region. A quantizer is uniquely defined by the codebook centroids and the distortion metric. The LBG algorithm is used to minimize the overall distortion of the quantizer by iteratively moving the centroids and computing the new distortion until the quantizer converges on a local minimum. Previous implementations of the LBG algorithm compute the centroid by adding all the vectors in the Voroni region and then dividing by the number of vectors. This is done iteratively on a sample of source data referred to as a training sequence.

Adaptive Vector Quantization is a procedure that simultaneously designs codebooks as the data is being encoded or quantized. This is done by computing the centroid as a recursive moving average where the

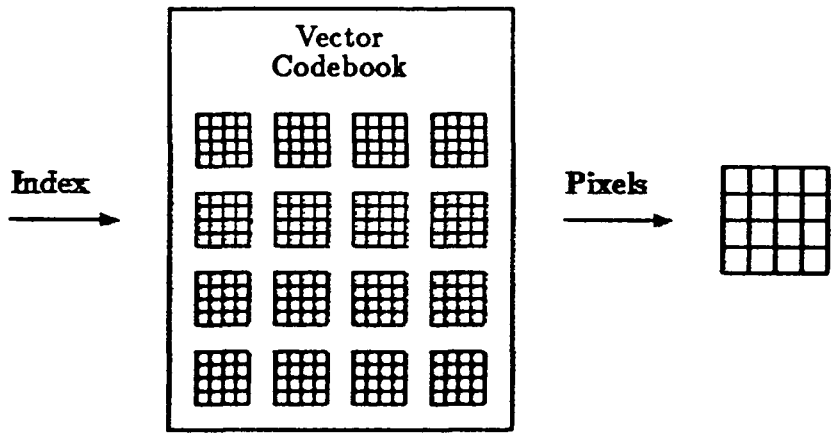
centroids move after every vector is encoded. When computing the centroid of a fixed set of vectors the resultant centroid is identical to the previous centroid calculation. This method of centroid calculation can be easily combined with VQ encoding techniques. The defined quantizer changes after every encoded vector by recursively updating the centroid of minimum distance which is the selected by the encoder. Since the quantizer is changing definition or states after every encoded vector, the decoder must now receive updates to the codebook. This is done as side information by multiplexing bits into the compressed source data. It is important to note that the quantizer converges in much the same way as the LBG algorithm converges. For stationary data sources the centroids will become fixed and the side information will not be necessary. For non-stationary sources the side information can be used to allow the quantizer to adapt to the data, thereby providing an Adaptive Vector Quantizer. Important issues to consider are the rate of convergence, start-up distortion or rate overhead, and tracking non-stationary sources. These issues will be addressed in a forthcoming publication.

ACKNOWLEDGEMENTS

This work was partially supported by Unisys Corporation
through the University of Utah Center for Communications Research.



Vector Quantization Encoding



Vector Quantization Decoding

Present Implementation of VQ

- Acquire data from sensor
- Design a codebook
- Implement a search technique

Acquire Data from the sensor

- Expensive
- Classified
- Not possible
- Poor representative
  (start over)

Codebook Design

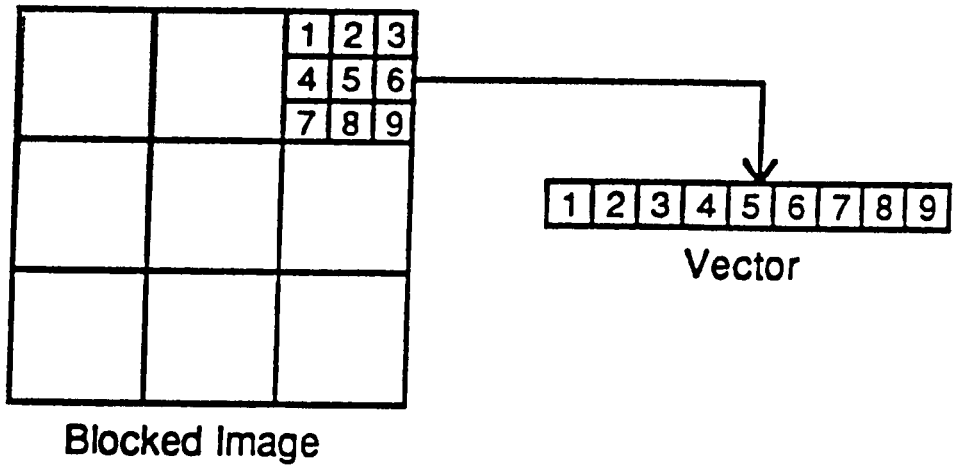
- Exhaustive Search
- Generalized Lloyd (LBG)
- K-means
- Simulated Annealing
- Pairwise Nearest Neighbor (PNN)

One Dimensional Lloyd's Algorithm

- Determine PDF of source
- Solve Equations
  - $x_j = \frac{y_j + y_{j-1}}{2} \qquad j = 2, ..., N$
  - $\int_{x_j}^{x_{j+1}} (x - y_j) p(x) dx = 0 \qquad j = 1, ..., N$

Generalized Lloyd using a training sequence

- Acquire training sequence
  (lots of samples of source data)
- Create vectors by grouping samples
  (maximize correlation)



- **Design codebook**

1. Initialize codebook

Place a set of quantization points in the vector space

2. Encode training sequence

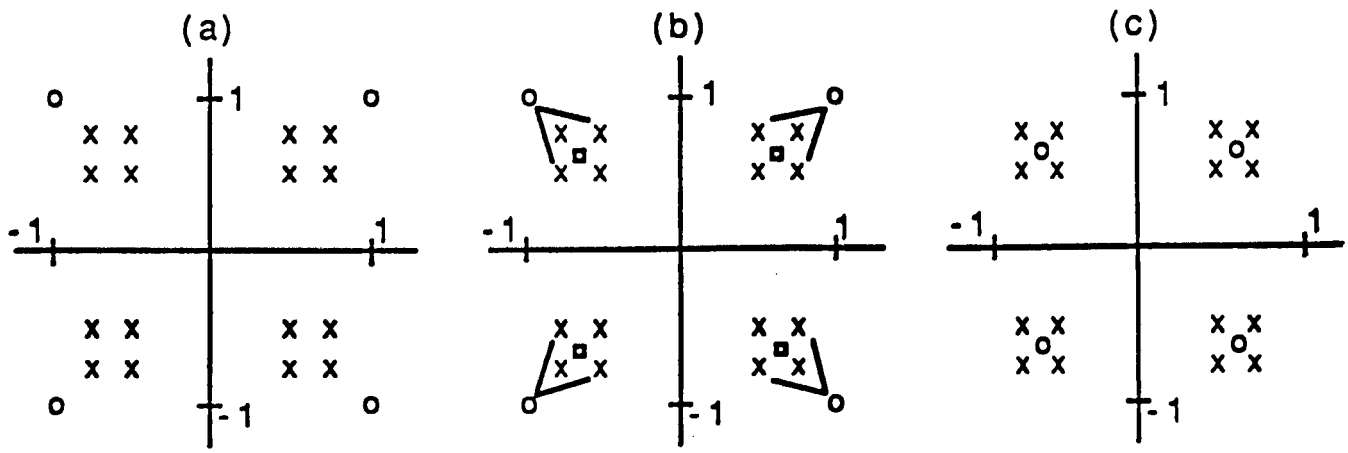
Assign each vector from the training sequence to the closest quantization point

3. Reassign codebook

Compute the centroid of each set of training sequence vectors assigned to a codebook vector and reassign the codebook to be these new centroids

4. Iterate 2 and 3

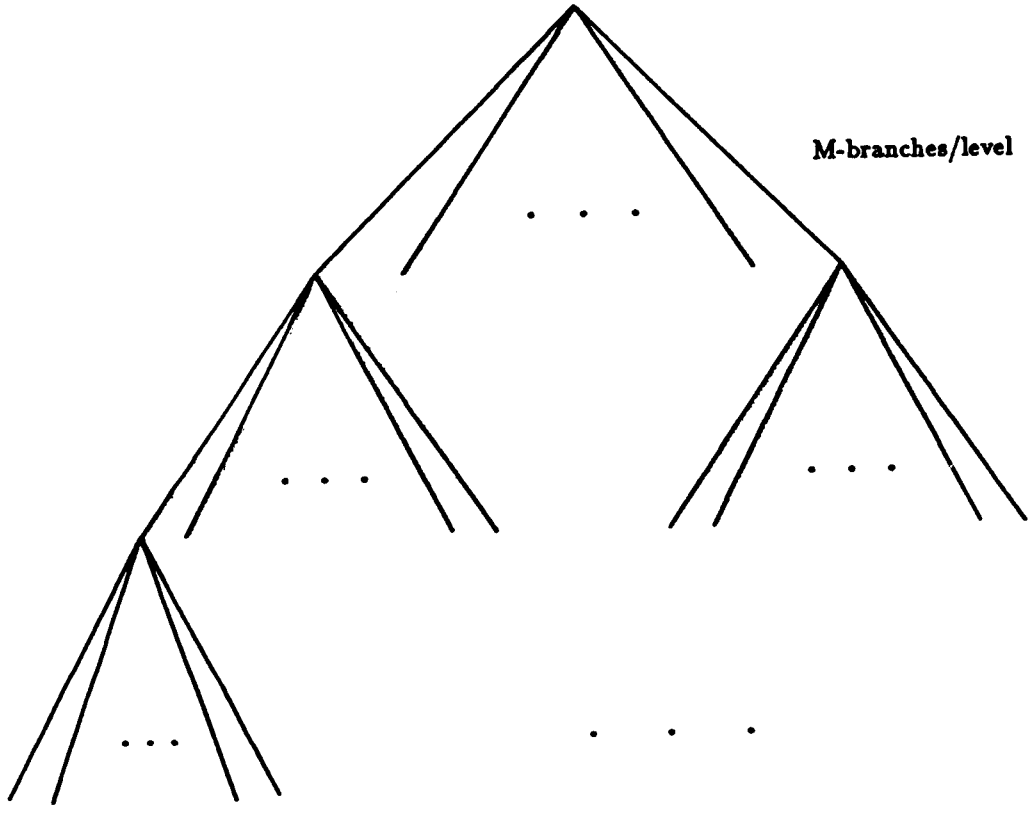
Iterate until no change (or minor change) to the overall distortion



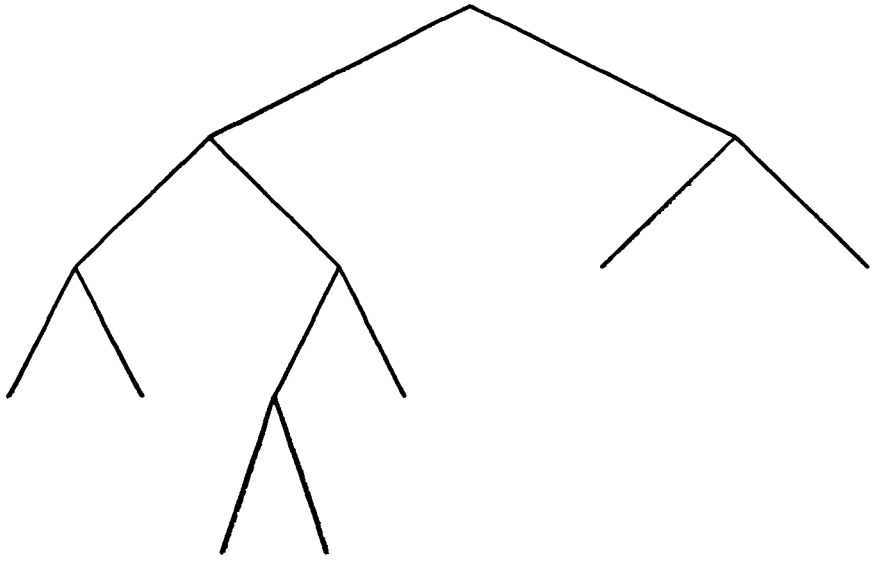
Encoding the Source Vectors Using Full Search

- Compute the quantization error (distortion) between the source vector and each vector in the codebook
- Replace the source vector with the index to the vector of minimum distortion

Tree



NON-UNIFORM BINARY TREE SEARCH



Each level computes $\vec{s} \cdot \vec{v}(j) \lesseqgtr T(j)$ where

- $\vec{v}(j) = \vec{C}_2(j) - \vec{C}_1(j)$
- $\vec{T}(j) = \dfrac{\|\vec{C}_2(j)\|^2 - \|\vec{C}_1(j)\|^2}{2}$

Adaptive Vector Quantization

- Combines principles of codebook design with encoding
- Requires no source samples of data to start (training sequence)
- Removes logistics problem of changing codebooks

Recursive Codebook Design

- Initialize codebook vectors
- Encode a source vector as before by looking at each entry in the codebook and choosing entry of minimum distortion
- Update the codebook vector after every encoded source vector
- Send Δchange as side information

Changing Codebook Values

- Codebook entries are the centroids of the Voroni region
- Centroid computation

  1. $C = \frac{1}{N} \Sigma_{i=1}^{N} x_i$
  2. $C_n = \frac{n-1}{n} C_{n-1} + \frac{1}{n} x_n \qquad n = 0, 1, ..., N$

- Codebook is a set of changing centroids

$$y_{n_1} = \frac{n_1-1}{n_1} y_{n_1-1} + \frac{1}{n_1} x_{n_1} \qquad n_1 = 0, 1, ...$$
$$y_{n_2} = \frac{n_2-1}{n_2} y_{n_2-1} + \frac{1}{n_2} x_{n_2} \qquad n_2 = 0, 1, ...$$
$$\vdots$$
$$y_{n_M} = \frac{n_M-1}{n_M} y_{n_M-1} + \frac{1}{n_M} x_{n_M} \qquad n_M = 0, 1, ...$$

- Centroids converge in much the same way as the Generalized Lloyd algorithm

Things to Consider

- Start-up
  - Increase rate
  - Increase distortion
  - Reset
- Convergence
- Register overflow
  - N counts
  - α divides
- Overhead for side information
- Performance
- Stationarity of source