Multimedia Communications

Vector Quantization

McMaster University

Vector Quantization

- By grouping source outputs together and encoding them we can extract the source structure and obtain efficient compression.
- Source outputs are grouped into blocks or vectors of length L.
- At the encoder and decoder there is a set of M, L-dimensional vectors called the code-book.
- Vectors in the code-book are called code vectors (levels) or output points.
- Each code vector is assigned a binary index.

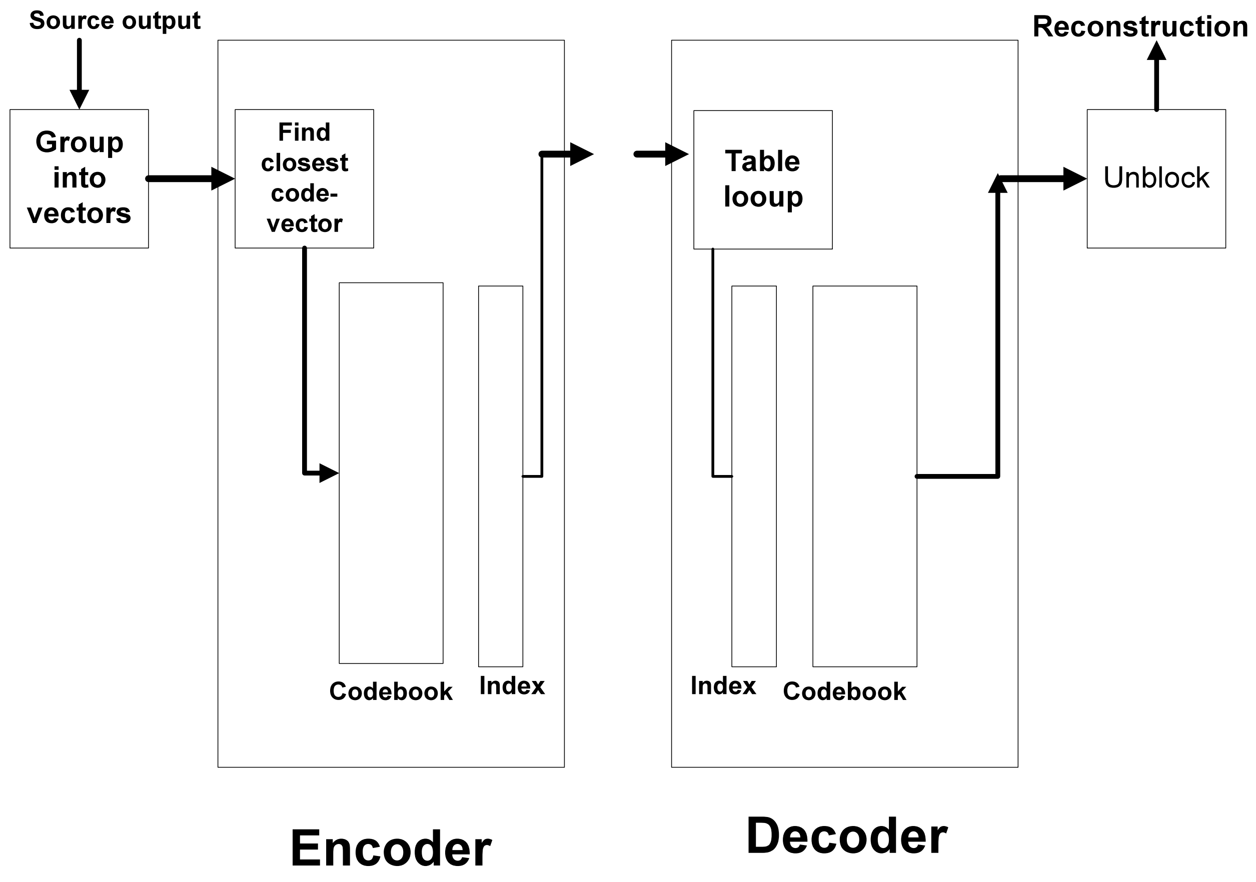
Vector Quantization

- Encoder: the input vector is compared to each of the code vectors to find the closest one.
- The binary index of the selected code vector is sent to decoder.
- Decoder has exactly the same codebook and can retrieve the code vector given the binary index.
- If the output of the VQ codebook is fixed-rate coded, then the rate is $r = (\text{Log}_2 M)/L$ bit per sample.
- Distortion measure: mean squared error $\|X - Y_j\|^2 \le \|X - Y_i\|^2$

$$Q(X) = Y_j$$

- An important problem in VQ is the design of the code-book

Vector Quantization



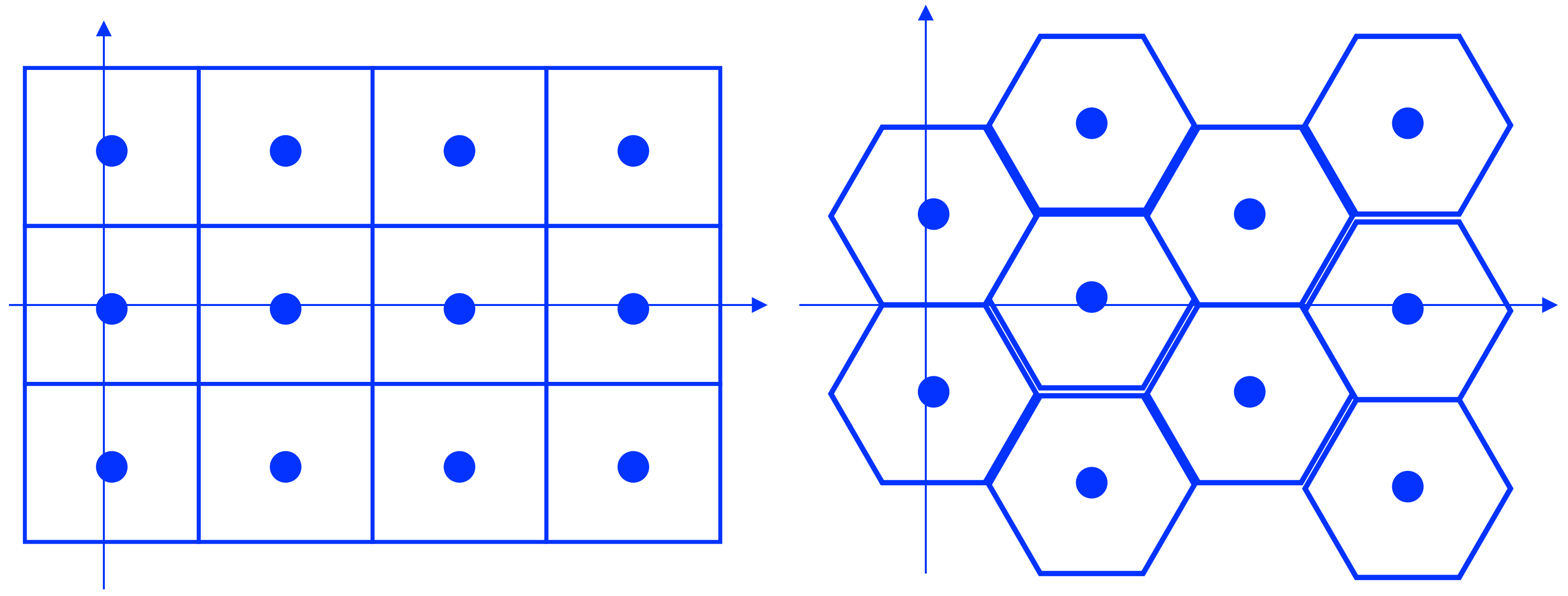
Vector Quantization

- Example: 2-D Quantizer
  - A map of the city that is divided into school districts.
  - Each school in its district represents the code vector.
  - Input: Location of child's residence.
  - Output: Rule by which each child is assigned to a school.

- Theorem: Given a signal vector (or data vector), no other coding technique exists that can outperform VQ [Shannon].

Vector Quantization: Advantages

- Exploit structure in the source samples
- More flexibility in terms of design



LBG Design Algorithm

1. Initialize Codebook (reconstruction vectors)
2. Find quantization regions $V_i^{(k)} = \{X : d(X,Y_i) < d(X,Y_j) \forall j \neq i\}$
3. Using the pdf compute the distortion D^k.

$$D^{(k)} = \sum_{i=1}^{M} \int_{V_i^k} \left\| X - Y_i^{(k)} \right\|^2 f_x(X)dX$$

4. If $((D^k - D^{k-1})/ D^k < \varepsilon$, stop otherwise continue
5. Increment k find new reconstruction vectors (centroid of regions) and go to step (2)

$$Y_i^{(k+1)} = \frac{\int_{V_i^k} X f_X(X)dX}{\int_{V_i^k} f_X(X)dX}$$

- Using the pdf is not practical. Most often training sets are used instead of pdf.

LBG Design Algorithm

- The practical algorithm is very similar to k-means (used in pattern recognition)

1. Initial set of reconstruction values $\{Y_i^{(0)}\}_{i=1}^{M}$ set of training vectors $\{X_n\}_{n=1}^{N}$
2. Assign each of the training vectors to one of the reconstruction vectors $V_i^{(k)} = \{X_n : d(X_n, Y_i) < d(X_n, Y_j) \forall j \neq i\}$
3. Compute average distortion D^k between training vectors and their representative reconstruction vector
4. If $(D^k - D^{k-1})/ D^k < \varepsilon$, stop otherwise continue
5. k=k+1. Find new reconstruction values (average of the elements of each quantization region)

LBG Design Algorithm

- Initial Codebook: The performance of the LBG algorithm depends heavily on the initial codebook.
- Splitting technique:
  - start with a single reconstruction vector (average of training set)
  - Obtain the initial code book of a two level VQ by including the output point of the one level VQ and a second vector which is obtained by adding a fixed perturbation vector to first vector
  - Use LBG algorithm to obtain the two level VQ
  - The initial four-level codebook consists of the two codebook vectors from the final codebook of two level VQ and two others obtained by adding the perturbation to the two codebook vectors.

LBG Design Algorithm

- Random selection (Hilbert)
  - Select the initial vectors randomly from the training set
- Pairwise Nearest Neighbor (PNN) method
  - Start with the set of training vectors
  - At each stage combine the two closest vectors into a single vector (their mean)
  - Continue until you get the required number of initial vectors
- Set the initial codebook to the set of reconstruction levels of scalar quantizer

Empty cell problem

- Sometimes no input vector gets assigned to an initial output vector (code vector)
- This is a problem because in order to update an output point, we need to take the average of the input vectors assigned to that output (inside the cell of that output)
- There is a danger that we will end up with an output that is never used
- A common approach is to remove an output point that has no inputs associated with it and replace it with a point from the quantization region with most training points.

Structured Vector Quantization

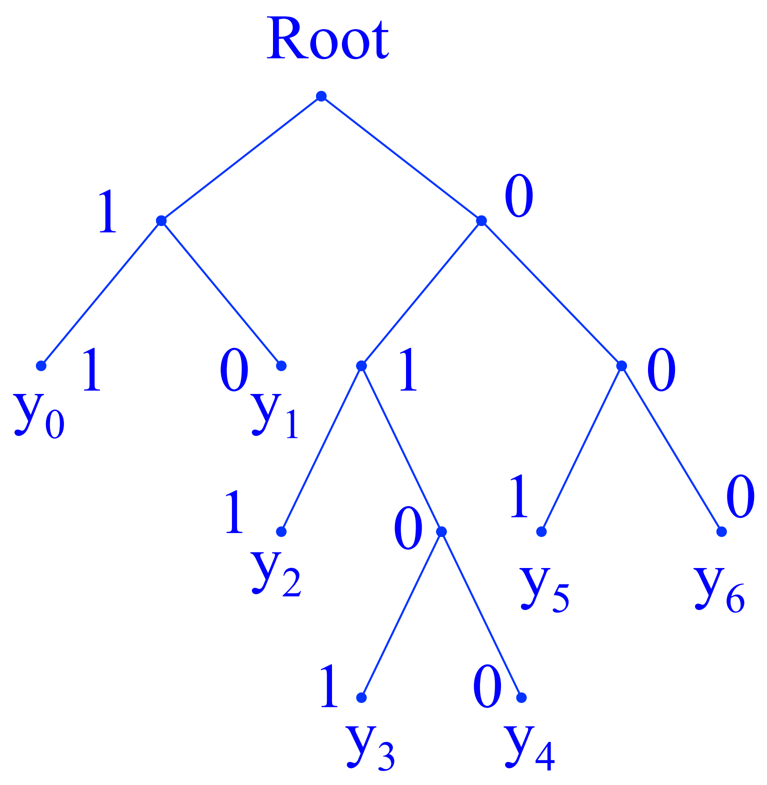
- Motivation (complexity): Let L be the dimension of the VQ. If R is the bit rate, then $L2^{RL}$ scalars need to be stored. Also, $L2^{RL}$ scalar distortion calculations are required.
- Solution: introduce some form of structure in the codebook and/or quantization process
- Substantial reduction in complexity (non-exponential dependence on dimension-rate product RL)
- Disadvantage: Inevitable loss in rate-distortion performance

Tree-structured vector quantization

- Divide the set of output points to two groups, g0 and g1.
- Assign to each group a test vector such that output points in each group are closer to test vector assigned to that group than the test vector assigned to the other group (v0 and v1)
- When we get an input vector we compare it with v0 and v1.
- Depending on the outcome the input vector is compared to the output points associated with the test vector and the other half is discarded.
- This process can be continued by dividing g0 into g00 and g01 and assigning test vectors v00 and v01 to each.

Tree Structured VQ (TSVQ)

- Tree Structured VQ (TSVQ)



Tree Structured VQ (TSVQ)

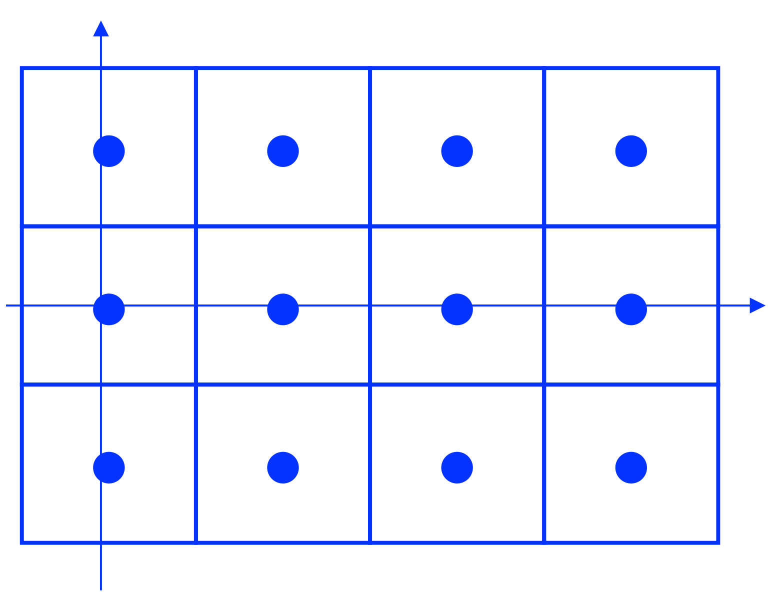
- The number of comparisons will be 2log M instead of M.
- Penalties:
  1. Possible increase in distortion
  2. TSVQ requires approx. twice the memory required by conventional VQ.
- How to design TSVQ?
  - Impose the structure after the VQ is designed
  - Design the VQ in the framework of the tree structure

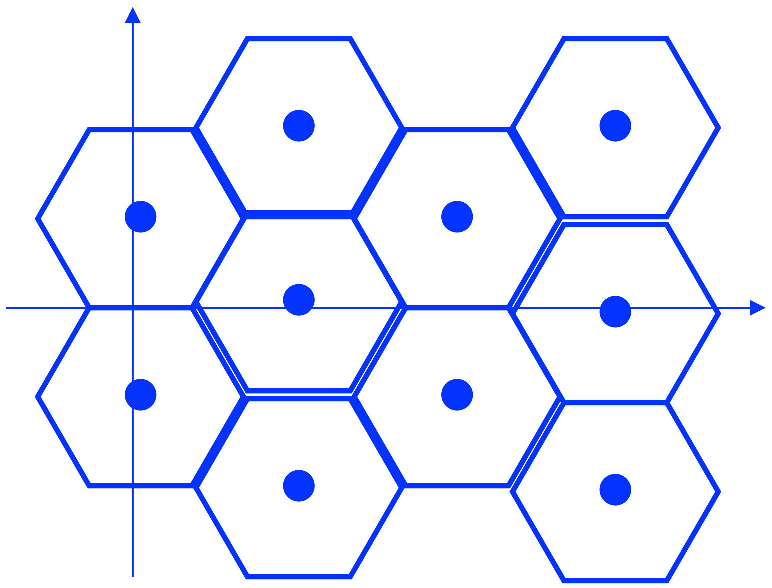
Tree Structured VQ (TSVQ)

- Obtain the average of all the training vectors, perturb it to obtain a second vector, and use these vectors to form a two-level VQ.
- Call the vectors v0 and v1 and the group of training set vectors that would be quantized to each as g0 and g1.
- Perturb v0 and v1 to get the initial vectors for a four-level VQ.
- Use g0 to design a two-level VQ and g1 to design the another two-level VQ.
- Label the vectors v00, v01, v10, v11.
- Split g0 using v00 and v01 into two groups g00, g01.
- Split g1 using v10 and v11 into two groups g10, g11.

Lattice VQ

- The codewords have a regular pattern





Lattice VQ

- Highly structured VQ (reduction in computation and memory)
- A VQ whose codebook consists of a subset of a lattice
- Simplest form: Cubic lattice
- If the source probability is confined to a bounded space, the lattice codebook should be confined to the same space.
- A one-dimensional cubic lattice is a uniform quantizer.

Lattice VQ

Definition: A Lattice in a k-dimensional space is the collection of all vectors of the form:

$$\sum_{i=1}^{n} m_i \underline{u}_i$$

where the m_i' s are arbitrary integers and the u_i' s form a linearly independent set of n<=k vectors.

McMaster University

Gain-shape VQ

- In some applications the dynamic range of the input is quite large.
- To represent the various vectors from the source, we need a very large codebook
- This requirement can be reduced by normalizing the source output vectors, then quantizing the normalized vector and the normalization factor separately
- Variations due to dynamic range is represented by the normalization factor (gain)
- This technique is called gain-shape VQ.

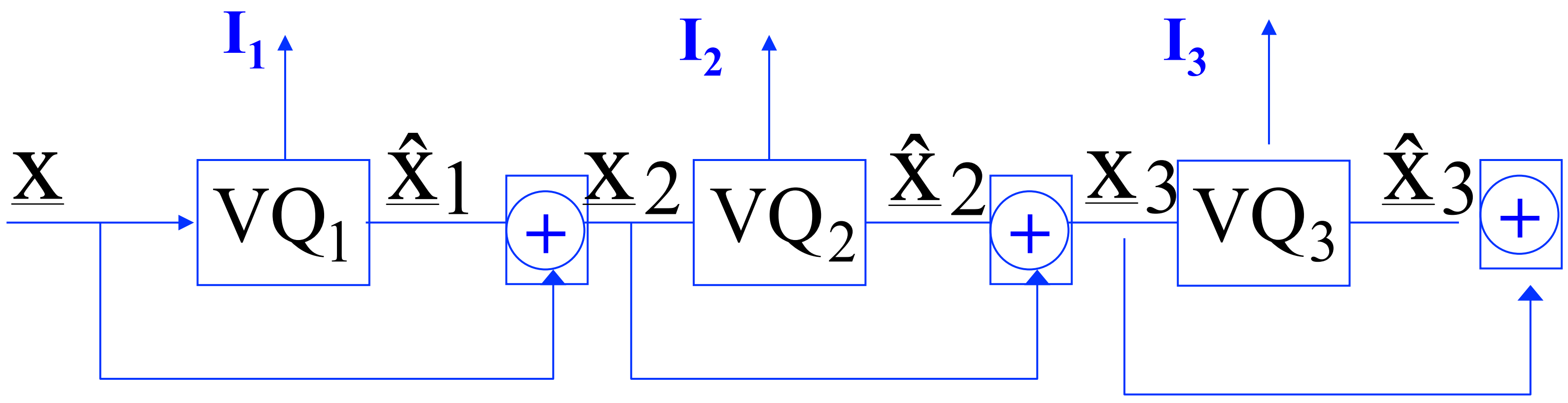
McMaster University

Pyramid VQ

- Pyramid VQ:
  - First find the value $r = \sum_{i=1}^{L} |x_i|$
  - This value (called gain) is quantized and transmitted to the receiver
  - The input is normailized by gain (called shape) and quantized using a single hyper-pyramid
  - Quantization of shape consists of finding the output point on the hyper-pyramid closest to the shape and finding a binary codeword for it.

Multistage VQ

- Multistage VQ: reduces both encoding complexity and the memory requirement for VQ
- Input is quantized in several stages



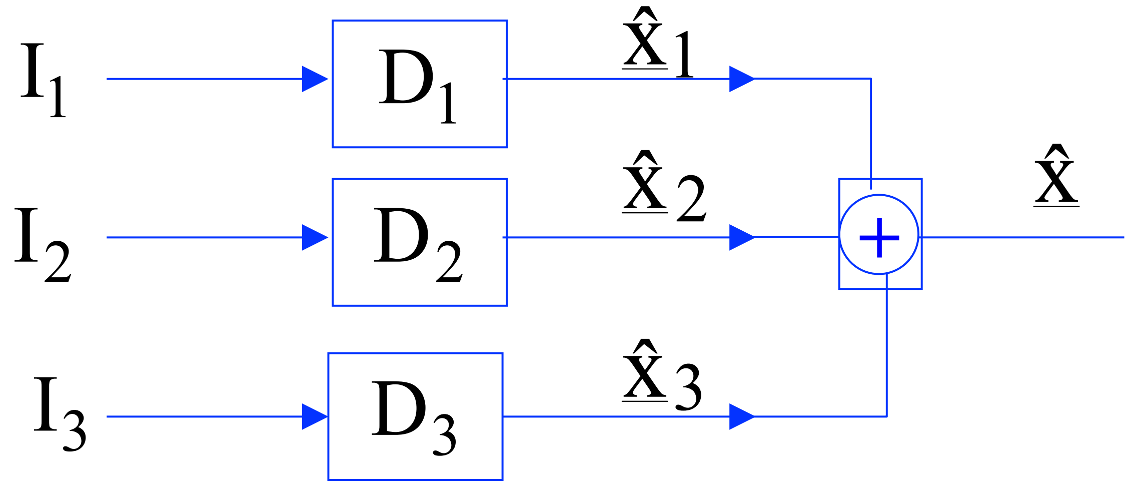
$$\underline{\hat{x}}_1 = Q_1(\underline{x})$$

$$\underline{\hat{x}}_2 = Q_2(x - \hat{x}_1)$$

$$\underline{\hat{x}}_3 = Q_3(x_2 - \hat{x}_2) = Q_3(x - (\hat{x}_1 + \hat{x}_2))$$

Multistage VQ

- If the size of codebook of the nth stage is L_n, then the effective size of the overall codebook is $L_1 x L_2 x \ldots x L_k$.
- We need to store only $L_1+L_2+\ldots+L_k$ vectors.
- For same rate, performance of this method is lower than LBG.



$$\hat{\underline{x}} = \hat{\underline{x}}_1 + \hat{\underline{x}}_2 + \hat{\underline{x}}_3$$

McMaster University

Classified VQ

- We can sometimes divide the source output into separate classes with different properties
- A separate VQ is designed for each class
- The encoder transmits both the label for the codebook and the label for the vector in the codebook