Efficient Methods for the Addressing/Decoding of a Lattice-based Fixed-rate, Entropy-coded Vector Quantizer*

S. Nikneshan and A. K. Khandani

Dept. of Elec. and Comp. Eng., University of Waterloo, Waterloo, Ont., N2L 3G1

Email: sasan@shannon.uwaterloo.ca, khandani@shannon.uwaterloo.ca

Abstract: In the quantization of a non-uniform source, the entropy coding of the quantizer output can result in a substantial decrease in bit rate. A straight-forward entropy coding scheme faces us with the problem of variable data rate. A solution in a space of dimensionality N is to select a subset of elements in the N-fold cartesian product of a scalar quantizer and represent them with code-words of the same length. For a memoryless source, a reasonable rule is to select the N-fold symbols with the lowest additive self-information. The *search/addressing* of this scheme can no longer be achieved independently along the one-D subspaces. Fortunately, the selected subset has a high degree of structure which can be used to substantially decrease the complexity. We discuss a method based on dynamic programming to facilitate the *search/addressing* operations. We build our recursive structure required for the dynamic programming in a hierarchy of steps. This results in several benefits over the conventional trellis-based approaches. Using this structure, we develop efficient rules (based on merging the states) to substantially reduce the search/addressing complexities while keeping the degradation negligible. We choose the quantizer points from a lattice resulting in a higher granular gain in comparison with simply using the Cartesian product of a set of scalar quantizers. We introduce a special class of lattices which have a low decoding complexity, and at the same time result in a noticeable granular gain.

1 Introduction

Consider the problem of quantizing a source with a nonuniform probability density function. In the N-D space, as N becomes large, the N-D probability density function concentrates in a region where the density function is almost uniform (typical set). In a fixed-rate entropy coding scheme, the typical set is selected as the subset of the N-D symbols to be quantized, and the corresponding quantizer partitions are represented with code-words of the same binary length. In this case, as some of the elements in the N-fold Cartesian product space are not allowed, the *search* for the quantizer partition (decoding) and also the corresponding *addressing, reconstruction* processes can no longer be achieved independently along the one-dimensional (one-D) subspaces.

We choose the elements of the quantizer from a lattice resulting in a higher quantization gain in comparison with simply using the Cartesian product of a set of scalar quantizers. We make use of a special class of lattices which have a low decoding complexity, and at the same time result in a noticeable quantization gain. We combine the procedure of the lattice decoding with that of quantizer shaping using hierarchical dynamic programming. In addition, by using appropriate partitioning and merging rules, we obtain sub-optimum schemes of very low complexity and small performance degradation.

2 Lattice-based quantization

A real lattice Λ is a discrete set of vectors in real Euclidean N-space, R^N, that forms a group under ordinary vector addition. *Fundamental region* of a lattice is a building block which when repeated many times fills the whole space with just one lattice point in each copy.

Assume that there are M quantizer points, say $Q_1, Q_2, \ldots, Q_M$, which have been chosen according to a specific rule in R^N. For a quantizer input x, which is an arbitrary point of R^N, the quantizer output is the point Q_i which has the smallest distance to x. In other words, the space R^N is partitioned into the Voronoi cells $V(Q_1), V(Q_2), \ldots$ around the Q_i's. If the input x belongs to $V(Q_i)$, the quantizer output will be Q_i.

A lattice quantizer is based on using the points of a lattice to partition the space into the quantization regions. In this case, the structure of the lattice is used to facilitate the quantization operation. In the present work, we introduce a special class of lattices which are constructed from the structure of the Hadamard matrix.

A square matrix with rank $L = 2^l$ is a Hadamard matrix if all of its elements are equal to 1 or -1, and the

*This work has been supported in part by the Information Technology Research Centre (ITRC) of Canada and in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

set of all its L rows and L columns form an orthogonal basis for the L-D space. We define a set of vectors $V = \{v_1, v_2, \ldots, v_L\}$ such that v_i represents the ith row of an $L \times L$ Hadamard matrix. The vector $v_i^c \in V^c$ (complement of the vector $v_i \in V$) is obtained by replacing all the 1's in v_i by -1, and vice versa.

Consider the set $W = V \cup V^c$ which is composed of $2L$ vectors in an L-D space. We define a new set called $W^{(i)}$ whose members are obtained by concatenating the members of W for i times, subject to:

1. Each vector can be concatenated with itself or with its complement.
2. All the possible combinations of vectors and its complements are allowed.

For example $w = (v_1, v_1^c, v_1, v_1^c)$ is a member of $W^{(4)}$. It should be mentioned that $W^{(i)}$ is of dimensionality iL, and has 2^i members. We use the elements of $W^{(i)}$ to define the fundamental region of an iL-D Lattices called Λ_L^i.

Assume that in the element of $W^{(i)}$, all the 1's are substituted by $e(even)$ and all -1's are substituted by $o(odd)$. Now assume that there are M threshold points along each dimension of the space. The lattice Λ_L^i is defined as the set of points in an $(i \times L)$-D space consisting of the points obtained from $W^{(i)}$ by replacing all the e's with the set of threshold points with an even index, and all the o's with the set of threshold points with an odd index. The final vector quantizer is selected as a subset of the points from $\Lambda_L^{N/L}$ composed of T elements, where each N-D point is represented by a code-word composed of $\lceil \log_2 T \rceil$ bits.

3 Basic structure

The core idea in the schemes of [1] is to use a state diagram with the transitions corresponding to the one-D symbols. This results in a trellis composed of N stages where N is the space dimensionality.The states s and $s+c$ in two successive stages are connected by a link corresponding to the one-D symbol(s) of cost c. Consequently, the states in the nth stage, $n=0,\ldots,N-1$, represent the accumulative cost over the set of the first n dimensions. The links connecting two successive stages are labeled by the corresponding one-D distortions. Then, the Viterbi algorithm is used to find the path of the minimum overall additive distortion through the trellis.

Unlike [1] which are based on a component-by-component analysis, we build our recursive structure in a hierarchy of levels where each level involves the Cartesian product of two lower dimensional subspaces. To explain this structure, let $F_N(C)$ denotes the set of the N-D points of the overall (additive) cost C (shell of cost C). We have the following *recursive* relationship:

$$F_N(C) = \bigcup [F_{N_1}(C_1) \otimes F_{N_2}(C_2)] \qquad (1)$$

where $\otimes$ denotes the Cartesian product, $N = N_1 + N_2$, and the union is computed over all the pairs (C_1, C_2) satisfying $C_1 + C_2 = C$. We are specially interested in the case that $N_1 = N_2 = N/2$. We refer to each Cartesian product element in Eq. 1 as a *cluster.*

As we are selecting the points from a lattice, we should impose another constraint on the Cartesian products of the shells when we are building the clusters in the next level of the hierarchy. These constraints are closely related to the recursive structure of the Hadamard matrix, and determine which subset of the elements in the Cartesian product of the lower dimensional subspaces are allowed.

3.1 Recursive decoding

For a given input vector $\mathbf{x}$, by decoding of a shell we mean the process of finding the element of the shell which has the minimum distance to $\mathbf{x}$. We make use of Eq. 1 to develop a recursive method for the decoding of the shells. To do this, the N-D input vector $\mathbf{x}$ is split into two parts $\mathbf{x}_1$ and $\mathbf{x}_2$ each of length $N/2$. Assume that the nearest vectors of the shells $F_{N/2}(C_1), F_{N/2}(C_2)$ to $\mathbf{x}_1, \mathbf{x}_2$ are equal to $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2$ with the distances d_1, d_2, respectively. The nearest vector of the cluster $F_{N/2}(C_1) \otimes F_{N/2}(C_2)$ to $\mathbf{x}$ is equal to $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2)$ with the distortion $d_1 + d_2$. The distortion of a shell is equal to the smallest of the distortions of its clusters. Note that if we know the distortion and the nearest vector for all the shells of the $N/2$-D subspaces, we can decode all the N-D shells. In other words one is able to decode an $N = 2^u$ dimensional space in u steps by starting from the one-D subspaces and progressing in a recursive way [2]. In this process, the space dimension at the k'th stage, $k = 0, \ldots, u-1$, of the hierarchy are grouped into 2^{u-k} groups which we call blocks. The structure of the lattice imposes some constraints on the way the shells are formed in the Cartesian product of the subspaces at the different levels of the hierarchy. This procedure is explained in the following:

We have N blocks at the first stage of the hierarchy where each of these blocks consists of two sub-blocks, called *e(even)* and *o(odd).* To build the shells and the sub-blocks in the next stage, two shells from different blocks can build a shell for the next stage, iff they are from the same sub-block or from sub-blocks which are complement of each other. For instance, there are four sub-blocks in the second level of the hierarchy which are labeled as *ee, oo, eo, oe.* The Cartesian product of the shells of *ee* sub-block and *oo* sub-block is allowed and result in the shells of the *eeoo* sub-block, while for example, the Cartesian product of the shells of *eo* and *oe* are not allowed.

If the rank of the Hadamard matrix is equal to $L = 2^l$, in the $(l+1)$th stage of the hierarchy, the merging rule changes. At this level, the complement sub-blocks are merged together, and as a result, the number of the sub-blocks remains the same as in the previous level. After this stage, there will be always L sub-blocks in each block in the remaining stages of the hierarchy.

One can also use the recursive structure of the shells to develop an algorithmic addressing, reconstruction procedures. The basic idea is that the addressing within each cluster can be achieved independently along its lower dimensional shells and sub-blocks. This comes from the same decomposition principle as discussed in [3], [4], [5], accompanied with some extra considerations due to the existence of the lattice structure.

4 State space quantization

The straight-forward approach is to assign an independent state to each possible value of cost at a given level. If two different combinations of costs result in the same value for their sum, we say that the the corresponding states have *merged* together. This is denoted as a *natural merge*. Let K denote the number of the distinct values of cost along a dimension. Even for a moderate value of K, the number of distinct states in N-D can be impractically large. The solution is to synthetically *aggregate* distinct states into a smaller number. This is denoted as the *state-space quantization.*

In the following, we propose a rule for the state-space quantization which is specially effective and results in a simple addressing procedure.

4.1 Aggregation on a sequential basis

Consider an $N = 2^u$-dimensional space and assume that there are $K_i = 2^{k_i}$ macro-shells of equal cardinality in each sub-blocks at each level of hierarchy. In the Cartesian product of two of the $N_i = 2^i$-D subspaces, we obtain 2^{2k_i} clusters of equal cardinality. The clusters are arranged in the order of increasing average costs. A number equal to $2^{2k_i - k_{i+1}}$ of subsequent clusters are aggregated into a higher level ($2N_i = N_{i+1}$-D) macro-shell. Then, the whole process is repeated recursively. The final subset is obtained by keeping some of the N-D clusters of the lowest average cost. Note that this whole operation is done just once and the result is stored for subsequent uses. In this case, the total number of states (macro-shells) in the i'th level of the hierarchy is equal to: $(N/2^i) \times 2^{k_i} = 2^{u+k_i-i}$ where $N/2^i$ is the number of 2^i-dimensional subspaces (blocks) involved in the i'th level.

Using macro-shells of *integral, equal bit rate* results in a specially simple addressing scheme. This is discussed in the following: Consider the case that the macro-shells in a given level of our hierarchy, say at dimensionality N', are composed of 2^{c_1} elements. Also, assume that a higher level macro-shell (dimensionality $2N'$) is obtained by aggregating 2^{c_2} clusters in the two-fold Cartesian product of the N'-D macro-shells. The addressing of the $2N'$-D macro-shells requires $2c_1 + c_2$ bits. The address of an $2N'$-D element is computed by concatenating the addresses of its constituent components in the N'-D macro-shells and concatenating the result with an additional c_2 bits which are selected as the label of the cluster within its corresponding $2N'$-D macro-shell.

4.2 Merging of clusters using a binary tree

Assume that there are 2^k macro-shells of equal cardinality at a given stage of our hierarchy. In the 2-fold Cartesian product space, we obtain 2^{2k} clusters which are merged into 2^l macro-shells of integer bit rate. Define $2^{-\ell_i}$ to be the fraction of the number of clusters in the ith macro-shell, $i = 0, \ldots, 2^l - 1$. The ℓ_i's satisfying $\sum_i 2^{-\ell_i} = 1$. A simple argument shows that the ℓ_i's can be selected as the lengths of different paths in any binary tree with $2^l - 1$ intermediate nodes (resulting in 2^l final nodes). As the number of such trees is usually quite small, one can use an exhaustive search to find the best tree for a specific tradeoff between complexity and performance. This configuration allows to use a set of prefix codes for the addressing of the macro-shells.

This nonuniform merging rule is applied in the $(u-2)$th stage (stage indexed by $u-3$) of the hierarchy. The corresponding merging rule for the $(u-1)$th stage is as follows: If there are an integral power of two of successive macro-shells with equal cardinality, these are merged into a single, larger macro-shell. One can also apply this rule successively several times.

4.2.1 Numerical results

In the following, we present some numerical results concerning the performance and the complexity of the proposed method. Figures (1), (2) represent the result of quantizing the 256 × 256 lenna image using the proposed methd as compared to using an optimized scalar quantizer (the dimensions of the quantizer are selected along the dimensions of the Discrete Cosine Transform). Tables (1) and (2) presents the performance of the proposed method in the quantization of a memoryless Gaussian source. In all cases, a sequence of 30000 source vectors is used to design the quantizer and a different sequence of the same length is used to measure the resulting performance. It is observed that the proposed method results in about 2 to 3 dB improvement with respect to the previously known methods at the price of a reasonable increase in the complexity.

References

[1] R. Laroia and N. Farvardin, "A structured fixed-rate vector quantizer derived from variable-length scalar quantizer—Part I: Memoryless sources," *IEEE Trans. Inform. Theory,* vol. IT-39, pp. 851–867, May 1993.

[2] A. K. Khandani, "A Hierarchical Dynamic Programming Approach to Fixed-rate, Entropy-Coded Quantization," *IEEE Trans. Inform. Theory,* vol. IT-42, pp. 1298–1303, July 1996.

[3] G. R. Lang and F. M. Longstaff, "A leech lattice modem," *IEEE J. Select. Areas Commun.,* vol. SAC-7, pp. 968–973, Aug. 1989.

[4] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal spaces—Part I: shell-addressed constellations," *IEEE Trans. Inform. Theory,* vol. IT–39, pp. 1799–1808, Nov. 1993.

[5] A. K. Khandani and P. Kabal, "Shaping multi-dimensional signal spaces—Part II: shell-addressed constellations," *IEEE Trans. Inform. Theory,* vol. IT–39, pp. 1809–1819, Nov. 1993.

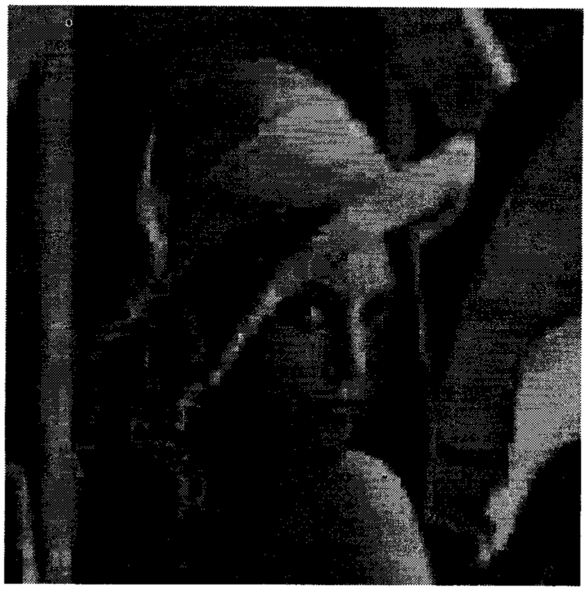

Figure 1: Lattice-based fixed-rate entropy-coded vector quantization of Lenna image, 0.5 bit/pixel, PSNR=27.14 dB

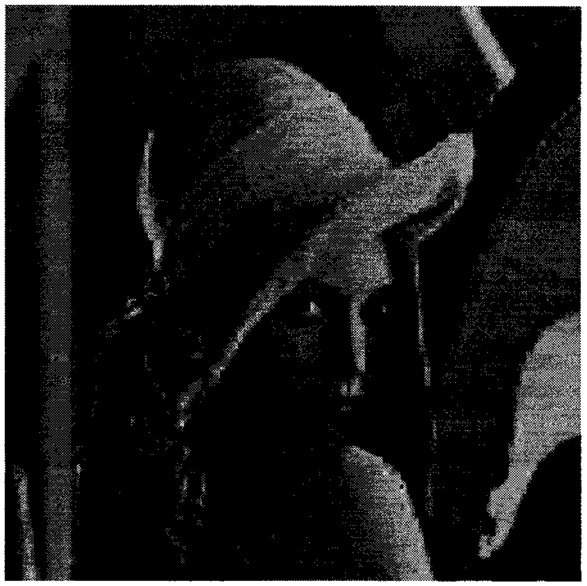

Figure 2: Scalar (LBG) Quantization of Lenna Image, 0.5 bit/pixel, PSNR=23.81 dB (bit allocation is achieved optimally).

Method	N	M	$(k_i, i = 0, u-1)$	R	M_{RAM}	M_{ROM}	$M_{\mathrm{RAM}} + M_{\mathrm{ROM}}$	Computation	SNR (dB)
Prop. method	16	8	$(2,4,5,7)$	1.5	4.8	24	28.8	10500	10.8
Ref. [2]	16	4	$(1,2,4,8)$	1.5	0.6	0.8	1.4 k	50	7.43
Ref. [1]	16	4	—	1.5	—	—	8 k	6×10^2	7.47
Prop. method	16	16	$(3,5,5,7)$	2.5	8.0	26	34 k	1760	16.18
Ref. [2]	16	8	$(2,4,5,8)$	2.5	1.0	2.0	3.0 k	220	12.91
Ref. [1]	16	8	—	2.5	—	—	21 k	2×10^3	13.00
Prop. method	16	32	$(3,5,6,6,10)$	3.5	64	127	191 k	45056	21.85
Ref. [2]	32	16	$(3,5,6,6,10)$	3.5	8	13	21 k	1100	18.7
Ref. [1]	32	16	—	3.5	—	—	300 k	1×10^4	18.8

Table 1: Comparison between the proposed method with the scheme of [1] and [2] for a memoryless Gaussian source. The quantities N, M and R are the space dimensionality, the number of points per dimension and the rate (in bits) per dimension, respectively. It is also assumed that L (rank of the Hadamard matrix) is 8 for all three conditions. The memory size is in byte (8 bits) per N dimensions and the computational complexity is the number of additions/comparisons per dimension.

Rate	N	M	L	$(k_i, i = 0, u-2)$	RAM	ROM	ROM+RAM	Computation	SNR(dB)
0.71	32	4	8	$(1,2,4,5)$	1.5	1.6	3.1	690	5.59
1.78	32	8	8	$(2,4,5,5)$	1.8	2.9	4.7	1190	11.04
2.78	32	16	8	$(3,4,5,5)$	2.2	2.9	5.1	1214	16.41

Table 2: Signal to noise ratio vs. the rate of the proposed scheme for $N = 32$ (using non-uniform merging).